Prioritized Multi-Objective Robot Control

Von der
Carl-Friedrich-Gauß-Fakultät
der Technischen Universität Carolo-Wilhelmina zu Braunschweig

zur Erlangung des Grades eines

Doktoringenieurs (Dr.-Ing.)

genehmigte Dissertation

von
Niels Jochen Dehio
geboren am 16.10.1988
in Frankfurt am Main

Eingereicht: 20. Juli 2018
Disputation: 11. Oktober 2018
1. Referent: Prof. Dr. Jochen J. Steil
2. Referent: Prof. Dr. Abderrahmane Kheddar
3. Referent: Dr. Michael Mistry

2018

Acknowledgement

Finishing the final draft, many people come to my mind, who in some way or another contributed to the completion of this thesis.

My deepest gratitude goes to my adviser Jochen Steil for the guidance of my thesis and for giving me the freedom to explore my own ideas while enjoying confidence in this respect. At University of Bielefeld he gave me the chance to participate in a European research project and when he took over the academic chair in Braunschweig he continued my supervision – offering me to join his new team which cannot be taken for granted.

I would like to extend my utmost gratitude to Michael Mistry for providing strong support and the great opportunity to spend five months in Edinburgh. Visiting his research team was inspiring and opened up new vistas.

Furthermore, I am also grateful to Felix Reinhart who provided me with relevant information about robotics research in the beginning of my Ph.D program and answered numerous questions with endless effort and care. With his help and motivation I improved my English skills a lot.

I wish to thank Joshua Smith for our intensive and fruitful discussions and for the joyful lab coding sessions. Special thanks to Pouya Mohammadi for his honest comments and many helpful suggestions. Sharing the office with him has always been a great pleasure. I owe much to Daniel Kubus for his good advice and for his willingness to work together with me on mathematical concepts.

I would like to thank also all my coauthors, colleagues and lab-mates for the encouraging atmosphere and a phenomenal research environment. It has always been a great pleasure to work with such an amazing team of people and their motivating feedback was invaluable in compiling this thesis.

I would not be where I am now without the continuous support and trust of my former excellent lecturers, school-teachers, friends, godparents and my family. Thanks for introducing me to the fascinating field of science, for spending your valuable time with me, and for encouraging me to go my own way.

Abstract – Prioritized Multi-Objective Robot Control

Niels Dehio

Humans are capable of pursuing multiple objectives simultaneously in everyday life, thereby flexibly prioritizing motion tasks and constraints among others. A major goal of current robotics research is to equip robots with similar adaptive behavior. This is a necessary prerequisite for future robotic applications in dynamic, unstructured and unpredictable or uncertain environments, for instance dangerous disaster scenarios.

Simultaneous tracking of multiple prioritized control objectives is of particular interest for highly redundant robots. Those systems have gained increasing attention during the last decades: Recent developments in robotics and mechatronics led to a new generation of robots with a high number of joints which improves versatility compared to standard industrial manipulators. Furthermore, advances in computational efficiency enable the treatment of multiple robots as a single multi-body system with many joints. The aim of this thesis is to endow such robot systems with the abilities to optimize priorities for possibly conflicting objectives, to smoothly rearrange such priorities and to decouple the objectives related to motion generation from the objective of simultaneous interaction with the environment through forces.

Executing a single task objective with robots has become increasingly proficient. However, flexibly prioritizing multiple objectives – such as introducing a novel task in addition to a preexisting set of tasks – is not yet solved adequately even though prioritization schemes have been studied extensively during the past decades. A commonly adopted approach is to tune priorities by hand once which then remain constant over time. In contrast to such constant prioritization, this thesis promotes a different view: A framework for automated learning soft priorities is introduced. Employing state-of-the-art optimization, different sets of priorities are evaluated offline and improved until the prioritized superposition of all underlying controllers satisfies a desired high-level goal.

As an alternative approach to soft prioritization with scalar weights, the well-known Stack-of-Tasks prioritization scheme relies on projectors with idempotence property and enforces strict priorities. The second part of this thesis performs a thorough formal analysis of projectors and proposes to smoothly shape between idempotent matrix operators. A novel prioritization scheme is presented with the help of this method, which generalizes previous approaches implementing either strict or soft priorities. It enables to insert new or remove outdated objectives if necessary and allows to rearrange priorities continuously online without inertia coupling while offering the ability to control the interference between objectives. This is an essential ability for robots to cope with changing situations or modified high-level goals.

Finally, underactuated robots subject to contact constraints are studied. A projection operator is derived which enables to decouple contact wrench control from constraint-consistent

motion generation for robots with passive or virtual joints. This is essential for controlling legged robots which are represented by a floating-base. Moreover, this thesis shows that the same principle holds for underactuated grasping scenarios. A fully-actuated multi-fingered robot hand is represented as an underactuated system when considering the grasped object as a rigid link connected by virtual joints. This representation captures the dynamics of the free-floating object and facilitates more precise manipulation.

The approach is validated with a large panel of experiments on various robot platforms both in simulation and the real-world. Priorities are learned in an automated fashion and smoothly rearranged during motion execution. Furthermore, motion generation is decoupled from contact wrench control. Application of the proposed approach to any torque-controlled stiff and potentially arborescent robot is straightforward given an accurate dynamic model. This thesis provides numerous tools that endows existing and future robot systems with ever-increasing degrees of freedom to create complex motion aiming to interact with their environment effectively.

Keywords: highly redundant cooperative multi-robot systems, multi-objective optimization, model-based torque-control, whole-body motion generation, dexterous grasping, underactuation

Zusammenfassung

Menschen sind in der Lage, mehrere Ziele gleichzeitig zu verfolgen und dabei flexibel mehrere Bewegungsaufgaben und -beschränkungen untereinander zu priorisieren. Ein wichtiges Ziel aktueller Robotik-Forschung ist es, Roboter mit einem ähnlich adaptiven Verhalten auszustatten. Dies ist von besonderem Interesse für hoch-redundante Roboter, welche in den letzten Dekaden verstärkte Aufmerksamkeit erfahren haben. Jüngste Entwicklungen in der Robotik und Mechatronik haben zu einer neuen Generation von Robotern mit hoher Anzahl von Gelenken geführt. Zusätzlich ermöglichen es Fortschritte in der Computertechnik, mehrere einzelne Roboter als ein großes Gesamt-System mit vielen Gelenken zu betrachten. Für zukünftige Roboter-Applikationen in dynamischen, unstrukturierten und gefährlichen Katastrophenszenarien sind das automatische Erlernen von Prioritäten, die Möglichkeit zum kontinuierlichen Anpassen solcher Prioritäten und das Entkoppeln von Zielen der Bewegungsgenerierung von gleichzeitiger Interaktion mit der Umwelt durch Kontaktkräfte eine notwendige Voraussetzung. Das Ziel dieser Dissertation ist es, Roboter mit ebendiesen Fähigkeiten auszustatten.

Während das Verfolgen eines einzelnen Zieles mit Robotern inzwischen gut möglich ist, gestaltet sich jedoch das flexible Priorisieren mehrerer Ziele bislang als schwierig – und dies obwohl an Methoden zur Priorisierung in den vergangen Jahrzenten intensiv geforscht wurde. Ein weit verbreiteter Ansatz ist das einmalige manuelle Bestimmen von Prioritäten, welche dann zeitlich konstant unverändert bleiben. Im Gegensatz dazu verfolgt diese Dissertation einen anderen Ansatz: Ein System zum automatischen Erlernen von Prioritäten wird eingeführt. Mit Hilfe von aktuellen Optimierungsmethoden werden unterschiedliche Priorisierungen in Simulation evaluiert und so lange verfeinert, bis das priorisierte Zusammenspiel aller zu Grunde liegenden Regler eine gewünschte übergeordnete Fähigkeit erfüllt. Als nächstes schlage ich einen geglätteten Übergang zwischen idempotenten Matrix-Operatoren vor, welcher auf einer gründlichen formellen Analyse von Projektionen basiert. Basierend auf dieser Methode wird ein neuartiges Priorisierungs-Verfahren aufgestellt, welches frühere Verfahren generalisiert, die entweder nur strikte oder nur weiche Prioritäten implementieren. Dieses Verfahren ermöglicht das Einfügen neuer und das Entfernen veralteter Ziele falls notwendig, und erlaubt Prioritäten kontinuierlich zur Laufzeit zu verändern. Dabei wird die Möglichkeit geboten, die Interferenz zwischen Zielen in einzelnen Dimensionen zu regeln. Zuletzt werden in dieser Dissertation unteraktuierte Roboter in Kontaktsituationen untersucht. Ich leite einen Projektions-operator her, welcher es Robotern mit passiven Gelenken ermöglicht, die Regelung von Kontaktkräften von der Bewegungsgenerierung zu entkoppeln. Dies ist essentiell für die Regelung mehr-beiniger Roboter, die keine feste Basis besitzen. Zusätzlich zeige ich auf, dass dasselbe Prinzip auch für Greif-Szenarien gilt, wenn ein virtueller Manipulator berücksichtigt wird, um die Objektdynamik zu kompensieren. Dies überführt einen viel-armigen Roboter in ein unteraktuiertes System mit virtuellen Gelenken. Der Ansatz wird validiert durch zahlreiche Experimente auf verschiedenen Roboterplattformen sowohl in Simulation als auch in der realen Welt. Die

Übertragung des Ansatzes auf beliebiege andere, Drehmoment-geregelte steife Roboter ist möglich, wenn ein akkurates dynamisches Modell gegeben ist.

Diese Dissertation beinhaltet eine Reihe an Werkzeugen, welche es aktuellen und zukünftigen Robotersystemen mit immer zahlreicheren Gelenken erlaubt, komplizierte Bewegungen auszuführen, mit dem Ziel, effektiv mit der Umwelt zu interagieren.

Schlüsselwörter: Redundante Robotersysteme, Mehrdimensionale Optimierungsmethoden, Modellbasierte Regelung, Bewegungsgenerierung

Contents

Chapter 1

Introduction

1.1 Motivation: The need for Prioritization between Objectives

Robots have recently become proficient in abilities such as flipping pancakes [1], or catching moving objects [2] to name just a few. Major research efforts have been spent in order to represent motion in joint-space, task-space or even lower-dimensional spaces [3, 4]. Following this direction, motion trajectories are partitioned into reusable segments of motion, resulting in a library of primitives. Subsequently, complex and continuous motion trajectories can be generated by *temporally sequencing* motion primitives [5, 6] to achieve a high-level goal. These approaches typically take place on a kinematic level and focus on the representation of motion. The control aspects are delegated to a dedicated *single-objective* control scheme.

Another popular paradigm to achieve high-level goals requiring complex motion generation is to *synthesize* multiple elementary control objectives *simultaneously* in real-time. However, it is not easy to ensure that all controllers work together in harmony. In this regard, *prioritization schemes* are imposed that aim to seek a compromise among all possibly conflicting objectives. Control objectives either represent constraints or are specified as motion tasks [7]. The authors in [8] argue that choosing between the terms "task" and "constraint" is only a matter of context. The latter perspective is adopted throughout this thesis. *Multi-objective* robot control is especially important for versatile, redundant robots, e.g. robots which inherently provide more degrees of freedom (DOFs) than necessary to fulfill the primary objective. This is the case for manipulators with seven joints when tracking a Cartesian end-effector trajectory. Humanoid robots provide even more redundancy. Many more control objectives can be achieved simultaneously when treating multiple cooperative robots and parts of the environment coupled though interaction as a single highly redundant multi-body system [9, 10].

How to devise a constant prioritization among objectives have been studied extensively over the last decades and constitutes the core of this thesis. Khatib proposed in [11] a hierarchical prioritization between two tasks by projection onto complementary spaces, namely the range and nullspace of the primary task. The execution of the primary task is guaranteed to be precise (if physically feasible) and decoupled from the secondary objective which again is handled within the nullspace. A classical example is tracking a Cartesian trajectory with the end-effector as primary task and as secondary objective utilizing the remaining redundancy to maintain the desired joint-space posture as close as possible without interfering with the trajectory tracking. When controlling highly redundant robots, multiple objectives can be superimposed within both subspaces to achieve a range of tasks simultaneously [12]. The Stack-of-Tasks (SoT) scheme extends the idea of projecting tasks onto subspaces to multiple stages for synthesizing several objectives, imposing a strict hierarchical prioritization between all tasks. Less important tasks are projected recursively onto the nullspace of higher-priority tasks, thereby ensuring decoupling. Constraints are included in the primary level of the hierarchy to guarantee their satisfaction.

SoT has been widely employed on different robot platforms, using both analytical [13–16] and optimization techniques [17–19].

Projection operators are idempotent by definition, which means that the result does not change when the operator is applied multiple times. Various works in SoT, however, replace projections with non-idempotent matrix operators that perform approximate projections. The resulting errors are typically accepted and not evaluated further, which holds in particular for the well-known damped least squares regularization [20]. This thesis refers to such approximations of projections as *shaped projections*, a term that was introduced in [21].

A drawback within such hierarchical schemes based on inverse kinematics or inverse dynamics is handling of irrelevant tasks which produce negligible small control commands. These tasks are blocking degrees of freedom (DOFs) which could potentially be better employed by other tasks assigned with lower priorities in the hierarchy. Sentis and Khatib [14] propose so-called "activation zones" to insert and remove tasks to or from the stack. This ability is also required when sequencing different tasks or constraints, for example when establishing a new contact. Deactivating tasks when not needed enables to forward the remaining redundancy to the next lower priority task. However, it was pointed out that task transitions caused by abruptly deactivating single tasks result in discontinuous control commands [22], [23].

The SoT scheme requires a strict task hierarchy which has to be designed by the developer in advance based on prior knowledge and has to stay constant over time. This scheme of hierarchical prioritization cannot accommodate flexibility with respect to changing priorities easily which is a necessary feature to react to sudden environmental changes that often occur in real-world scenarios. Swapping the order of priorities within SoT abruptly during motion execution could result in discontinuities in the control law [22], [23], even when tracking smooth and physically feasible tasks. It is therefore a persistent research question how to smoothly rearrange a given prioritization at runtime while limiting the derivative of control commands. Mathematically speaking, the main issue is how to perform a continuous and interpretable transition between two different idempotent projectors. This thesis refers to such a transition as *projection shaping* because intermediate matrix operators are usually not idempotent.

Soft prioritization between objectives without utilizing nullspace projections is an alternative approach to the strict SoT scheme. In this spirit, a weighted sum of torques has been proposed in [24–26]. Introducing a set of scalar weights in the torque superposition which represents priorities is flexible with respect to priority rearrangement by continuously interpolating weights. This advantage comes at the cost of selecting suitable weights which is non-trivial. Given a high-level goal, it is crucial to find a suitable set of priorities to achieve an efficient interaction among the different objectives.

The challenges described above are often discussed for motion control in free space with fixed-base robots. Robot control in multi-contact situation comprises an additional control objective besides motion generation. Contact wrenches shall be generated, for example, to maintain a unilateral contact. This aspect is crucial for floating-base robots such as quadrupeds and humanoids. Legged robots are inherently underactuated and contacts are employed to resolve underactuation. The "Projected Inverse Dynamics Control" (PIDC) approach [27,28] decouples two control objectives for constraint-consistent motion generation and realizing desired contact wrenches by projection onto orthogonal subspaces. This approach works well for fully-actuated robots, but application to underactuated robots is non-trivial. [29] and [30] attempt to solve this problem, however requiring optimization techniques. In contrast to those methods, an analytical solution to project the reference torques onto the actuated joints is highly desirable.

This thesis assumes that a high-level goal, comprising multiple elementary control objectives, is always specified. For example, in [31] human operators teleoperate a humanoid diving robot, whose behavior arises from a multitude of simultaneous motion tasks. As an alternative to instructions given by a human, [32] provides a framework for autonomous cognitive reasoning to obtain such high-level goals.

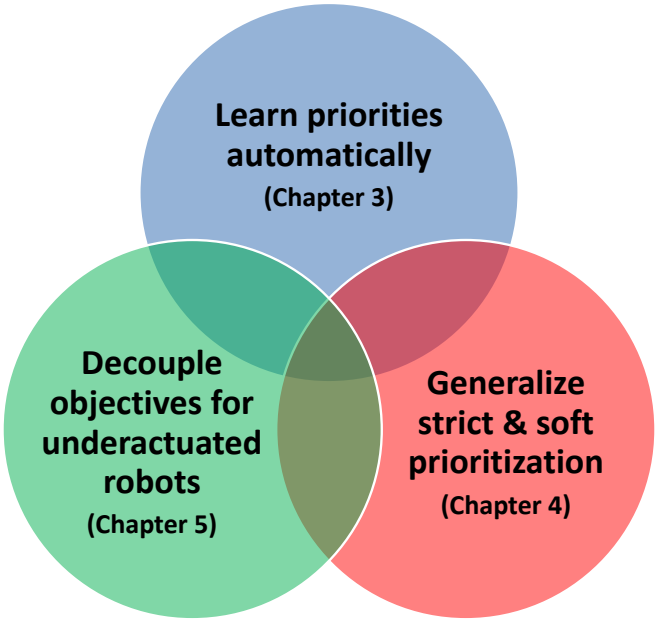


Figure 1.1: Synthesizing multiple prioritized objectives to create complex robot motion is discussed in this thesis. Three main problems are solved for highly redundant robots: The endeavor starts in chapter 3 with a framework for automated learning soft priorities. Next, soft and strict prioritization approaches are generalized into a novel control scheme in chapter 4. Chapter 5 focuses on underactuated robots in multi-contact situations. The contact wrench control objective is analytically decoupled from the motion generation tasks. The proposed approach in this thesis facilitates control of existing and future robot systems for complex motion in challenging environment.

1.2 Main Contributions and Goal of the Thesis

Simultaneous tracking of multiple prioritized control objectives is a fundamental capability for creating complex motion in dynamic, unstructured and unpredictable or uncertain environments. During the last decades, major research efforts have been spent in order to devise sophisticated prioritization schemes. In spite of the successive achievements, there are still a number of shortcomings which prevent full application in real-world applications and motivated this dissertation. The goal of this thesis is to overcome the difficulties mentioned above. The proposed approach is intended as next step towards adaptive and complex robot control in challenging environments. Considering multi-objective prioritization schemes for highly redundant torque-controlled robots, three persistent research questions are:

1. How to devise priorities among various control objectives?
2. How to combine and generalize advantages from both soft and strict prioritization?
3. How to decouple objectives for underactuated robots subject to contact constraints?

To address these questions, this thesis makes three major contributions (cf. Fig. 1.1), thereby enhancing flexible prioritized multi-objective robot control: To deal with the difficulties in selecting suitable priorities by hand the first idea is to treat the scalar priorities in the soft prioritization scheme as policy parameters and apply an stochastic policy search algorithm. Policy improvement with respect to a cost function automatically tunes the prioritization between tasks for different high-level goals. It is then experimentally verified, that continuous task-priority rearrangement can easily be implemented with the soft prioritization approach.

As the second contribution, a multi-objective prioritization scheme is introduced which generalizes and combines benefits from both strict and soft prioritization schemes. It allows shaping both operational space tasks and nullspace projections simultaneously and thereby consistently manages and adapts interference between tasks. To this aim, first shaping of operational space tasks and projections is formalized as a means to devise meaningful shaping operators analytically. Next, damped least squares regularization is reformulated as a shaped projector. In a final step based on rigorous formal analysis, a coherent real-time control scheme for continuous priority rearrangement between any subset of all tasks with dynamic consistency is proposed and experimentally validated.

The third contribution advances the Projected Inverse Dynamics Control (PIDC) scheme for underactuated robots subject to contact constraints. An analytic solution is derived to project the desired contact wrench control torques onto the actuated joints without interfering with motion generation tasks. Furthermore, it is proposed to treat a grasped object as an additional link virtually attached to the robot. On the one hand, this representation casts a fully-actuated multi-arm robot into a higher-dimensional and more complex underactuated system. On the other hand, this representation includes the object dynamics in the robot model and facilitates precise manipulation. The results show that modeling and controlling a multi-leg robot is equivalent to the proposed approach for a multi-arm robot. Real robot experiments demonstrate impedance-based manipulation of heavy objects in a force-closed grasp with a robot setup consisting of four cooperative KUKA LWR IV+ manipulators. Furthermore, the same controller is executed on the floating-base quadruped ANYmal standing on a slope ramp.

This thesis is organized as follows. After related works on multi-objective motion generation in free space as well as robot control in multi-contact situation have been presented in chapter 2, the automatic tuning of soft priorities is proposed in chapter 3. Next, a novel control scheme generalizing both soft and strict prioritization is introduced in chapter 4. Chapter 5 extends the PIDC approach to the underactuated case analytically. Decoupling of contact wrench control and motion generation is demonstrated for a floating-base quadruped and a four-fingered robot hand grasping a free-floating object. Finally, the main contributions of this thesis and its results are reviewed in chapter 6.

Chapter 2

Background & Related Research

This chapter first provides a broad overview of modeling robot kinematics and dynamics. Next, multi-objective robot control in the absence of contact constraints is discussed, and finally, the additional objective of contact wrench control is incorporated for multi-contact situations.

2.1 Robot Kinematics and Dynamics

In the following, a fixed-base robot with rigid links and D joints is considered. The kinematic relationship between joint velocities $\dot{\mathbf{q}} \in \mathbb{R}^D$ and the velocity $\dot{\mathbf{x}} \in \mathbb{R}^H$ of an H-dimensional operational point in task-space, e.g. the end-effector in Cartesian space, is given by the Jacobian $\mathbf{J} \in \mathbb{R}^{H \times D}$

$$\mathbf{J}\,\dot{\mathbf{q}} = \dot{\mathbf{x}}, \tag{2.1}$$

and its time derivative yields

$$\mathbf{J}\,\ddot{\mathbf{q}} + \dot{\mathbf{J}}\,\dot{\mathbf{q}} = \ddot{\mathbf{x}}. \tag{2.2}$$

The operational space formulation additionally relates task-space wrench $\mathbf{F} \in \mathbb{R}^H$ and joint torques $\boldsymbol{\tau} \in \mathbb{R}^D$ via the Jacobian transpose

$$\mathbf{J}^T\,\mathbf{F} = \boldsymbol{\tau}. \tag{2.3}$$

A robot is referred to be redundant with respect to control of an operational point iff $H < D$. In this case, the Jacobian $\mathbf{J}$ is a non-square matrix and is not invertible. Instead, the concept of pseudoinverse is typically employed [15] when rearranging the previous equations Eq. 2.1, Eq. 2.2 and Eq. 2.3 to compute the joint velocities, joint accelerations and end-effector wrench. Neglecting singular configurations assuming that the Jacobian $\mathbf{J}$ is always full-rank the (right) pseudoinverse (also referred to as generalized inverse) of $\mathbf{J}$ is given by

$$\mathbf{J}^{\mathbf{W},+} = \mathbf{W}\mathbf{J}^T(\mathbf{J}\,\mathbf{W}\,\mathbf{J}^T)^{-1} \text{ with } \mathbf{J}\mathbf{J}^{\mathbf{W},+} = \mathbf{I}, \tag{2.4}$$

where superscript indicates the positive definite weighting matrix $\mathbf{W} \in \mathbb{R}^{D \times D}$. The pseudoinverse is called Moore-Penrose inverse for $\mathbf{W} = \mathbf{I}$. In that case $\mathbf{J}^+ = \mathbf{J}^T(\mathbf{J}\mathbf{J}^T)^{-1}$ satisfies the following four Penrose equations:

$$\mathbf{J}\mathbf{J}^+\mathbf{J} = \mathbf{J} \text{ and } \mathbf{J}^+\mathbf{J}\mathbf{J}^+ = \mathbf{J}^+ \text{ and } \mathbf{J}\mathbf{J}^+ = (\mathbf{J}\mathbf{J}^+)^T \text{ and } \mathbf{J}^+\mathbf{J} = (\mathbf{J}^+\mathbf{J})^T. \tag{2.5}$$

For redundant robots there exist infinite solutions to compute joint velocities, accelerations and end-effector wrench. This manifold is reflected in the freedom of choosing a weighting matrix when computing the pseudoinverse:

$$\dot{\mathbf{q}} = \mathbf{J}^{\mathbf{W},+}\dot{\mathbf{x}} = \mathbf{W}\mathbf{J}^T(\mathbf{J}\mathbf{W}\mathbf{J}^T)^{-1}\,\dot{\mathbf{x}} \tag{2.6}$$

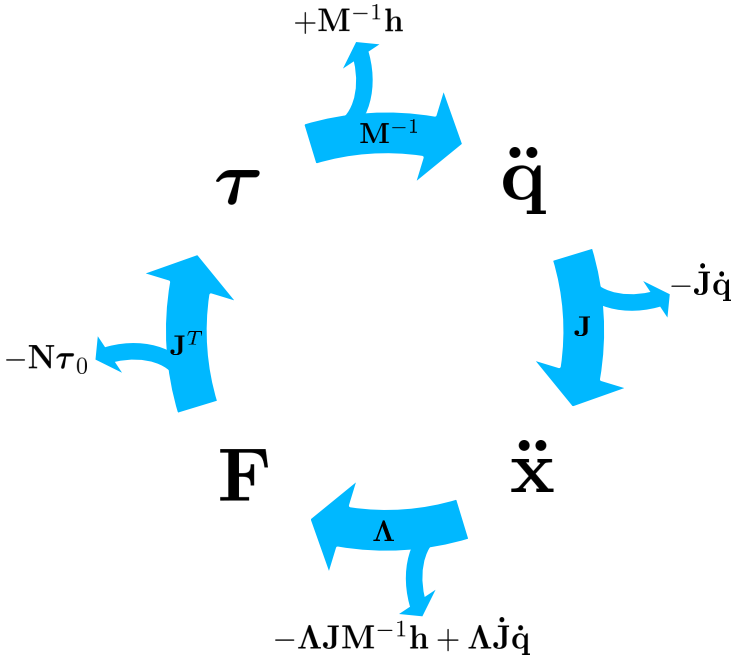


Figure 2.1: The four main equations for robot kinematics and dynamics in joint- and task-space that constitute an endomorphism: Eq. 2.2, Eq. 2.3, Eq. 2.9 and Eq. 2.11.

$$\ddot{\mathbf{q}} = \mathbf{J}^{\mathbf{W},+}(\ddot{\mathbf{x}} - \dot{\mathbf{J}}\,\dot{\mathbf{q}}) = \mathbf{W}\mathbf{J}^T(\mathbf{J}\mathbf{W}\mathbf{J}^T)^{-1}(\ddot{\mathbf{x}} - \dot{\mathbf{J}}\,\dot{\mathbf{q}}) \tag{2.7}$$

$$\mathbf{F} = (\mathbf{J}^{\mathbf{W},+})^T\boldsymbol{\tau} = (\mathbf{J}\mathbf{W}\mathbf{J}^T)^{-1}\mathbf{J}\mathbf{W}\boldsymbol{\tau} \tag{2.8}$$

These equations consider only the kinematic coupling between joints, neglecting dynamic properties of the links: mass, Center-of-Mass (CoM) and mass distribution. The (inverse) dynamics express the dynamic coupling between joints

$$\boldsymbol{\tau} = \mathbf{M}\ddot{\mathbf{q}} + \mathbf{h} + \mathbf{J}_c^T\mathbf{F}_c\ , \tag{2.9}$$

with symmetric positive definite joint-space inertia $\mathbf{M} \in \mathbb{R}^{D\times D}$, representing the robot's resistance to change in velocity, and compensation for gravity and Coriolis in joint-space $\mathbf{h} \in \mathbb{R}^D$. $\boldsymbol{\tau} \in \mathbb{R}^D$ describes joint torques and $\ddot{\mathbf{q}}$ accelerations in joint-space. Disturbances, e.g. due to human interaction or collisions with obstacles, are represented by an external contact wrench $\mathbf{F}_c$ and the Jacobian $\mathbf{J}_c$ associated with the contact. Both $\mathbf{M}$ and $\mathbf{h}$ are typically computed numerically with Newton-Euler recursion based on the dynamic properties of the links and the current joint configuration. In this thesis, non-linear effects such as structural joint and link flexibility, temperature effects, joint friction and measurement noise or delay are not considered for simplicity.

The relation between joint torques $\boldsymbol{\tau}$ and the acceleration $\ddot{\mathbf{x}} \in \mathbb{R}^H$ of an operational point is derived in the following. The robot's joint-space inertia matrix can be mapped onto task-space employing the corresponding Jacobian, resulting in the task-space inertia matrix $\boldsymbol{\Lambda} \in \mathbb{R}^{H\times H}$ with $\boldsymbol{\Lambda} = (\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T)^{-1}$. Hence, the joint-space dynamics Eq. 2.9 are mapped onto task-space by left-multiplication with the weighted projector onto the range[1] given by the term

[1]The theory of projection operators is explained in more detail below.

$\mathbf{J}^T(\mathbf{J}^{\mathbf{M}^{-1},+})^T = \mathbf{J}^T(\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T)^{-1}\mathbf{J}\mathbf{M}^{-1} = \mathbf{J}^T\mathbf{\Lambda}\mathbf{J}\mathbf{M}^{-1}$ resulting in

$$\begin{aligned}\mathbf{J}^T(\mathbf{J}^{\mathbf{M}^{-1},+})^T\boldsymbol{\tau} &= \mathbf{J}^T\mathbf{\Lambda}\mathbf{J}\mathbf{M}^{-1}(\mathbf{M}\ddot{\mathbf{q}} + \mathbf{h} + \mathbf{J}_c^T\mathbf{F}_c)\\ \mathbf{J}^T\mathbf{F} &= \mathbf{J}^T\mathbf{\Lambda}(\mathbf{J}\ddot{\mathbf{q}} + \mathbf{J}\mathbf{M}^{-1}\mathbf{h} + \mathbf{J}\mathbf{M}^{-1}\mathbf{J}_c^T\mathbf{F}_c)\end{aligned} \tag{2.10}$$

Substituting $\mathbf{J}\,\ddot{\mathbf{q}} = \ddot{\mathbf{x}} - \dot{\mathbf{J}}\,\dot{\mathbf{q}}$ in Eq. 2.2 results into the robot dynamics in task-space

$$\boldsymbol{\tau} = \mathbf{J}^T\mathbf{\Lambda}(\ddot{\mathbf{x}} - \dot{\mathbf{J}}\,\dot{\mathbf{q}} + \mathbf{J}\mathbf{M}^{-1}\mathbf{h} + \mathbf{J}\mathbf{M}^{-1}\mathbf{J}_c^T\mathbf{F}_c). \tag{2.11}$$

This formulation treats the robot as a single rigid object with the CoM located in the chosen operational point.

Note that the derivation of a Jacobian related to a specific operational space task is not always trivial because there is not always a direct relationship between the velocity in operational space and the joint-space. For example, [33] proposed recently a Jacobian representing the velocity mapping between Centroidal momentum and joint-space as well as the Jacobian for the Capture Point task which is beneficial for balancing with legged robots.

Fig. 2.1 visualizes the relationship of the main equations describing robot kinematics and dynamics in joint- and task-space, constituting an endomorphism.

2.2 Multi-Objective Robot Control in Free Space

Unencumbered motion generation in free space is less complex compared to robot control in contact situation which will be explained in the subsequent section. This thesis focuses on torque-control, even though similar concepts can also be employed for position-control.

2.2.1 Controlling a single Objective

A desired joint-space trajectory represented by reference joint accelerations $\ddot{\mathbf{q}}_{\text{des}}$ is classically tracked employing a proportional-derivative (PD) feedback control law to correct for errors. Based on Eq. 2.9, the total torque command $\boldsymbol{\tau}_{\text{cmd}} \in \mathbb{R}^D$ send to the robot is then

$$\boldsymbol{\tau}_{\text{cmd}} = \mathbf{M}\left(\ddot{\mathbf{q}}_{\text{des}} + \mathbf{K}\left(\mathbf{q}_{\text{des}} - \mathbf{q}_{\text{cur}}\right) + \mathbf{D}\left(\dot{\mathbf{q}}_{\text{des}} - \dot{\mathbf{q}}_{\text{cur}}\right)\right) + \mathbf{h}, \tag{2.12}$$

with positive definite gains matrices $\mathbf{K}$ and $\mathbf{D} \in \mathbb{R}^{D\times D}$. Instead controlling desired reference accelerations in task-space $\ddot{\mathbf{x}}_{\text{des}}$ the classical PD-control law is based on Eq. 2.11 with positive definite gains matrices $\mathbf{K}$ and $\mathbf{D} \in \mathbb{R}^{H\times H}$

$$\boldsymbol{\tau}_{\text{cmd}} = \mathbf{J}^T\mathbf{\Lambda}\left(\ddot{\mathbf{x}}_{\text{des}} + \mathbf{K}\left(\mathbf{x}_{\text{des}} - \mathbf{x}_{\text{cur}}\right) + \mathbf{D}\left(\dot{\mathbf{x}}_{\text{des}} - \dot{\mathbf{x}}_{\text{cur}}\right) - \dot{\mathbf{J}}\dot{\mathbf{q}} + \mathbf{J}\mathbf{M}^{-1}\mathbf{h}\right). \tag{2.13}$$

When applying classical PD-control the exact behavior in case of deviations from the desired trajectory is not clear. Impedance control allows realizing an explicit behavior of the system with respect to an external disturbance [34]. In more detail, the chosen operational point behaves like a mechanical mass-spring-damper system, transforming deviations $\widetilde{\mathbf{x}} = \mathbf{x}_{\text{des}} - \mathbf{x}_{\text{cur}}$ from the desired pose into wrenches

$$\mathbf{F}_{ext} = \mathbf{\Lambda}_{\text{des}}\ddot{\widetilde{\mathbf{x}}} + \mathbf{K}_{\text{des}}\widetilde{\mathbf{x}} + \mathbf{D}_{\text{des}}\dot{\widetilde{\mathbf{x}}}, \tag{2.14}$$

with a desired task-space inertia $\mathbf{\Lambda}_{\text{des}}$, a desired damping $\mathbf{K}_{\text{des}}$ and stiffness matrix $\mathbf{D}_{\text{des}}$. The feedback control law

$$\boldsymbol{\tau}_{\text{cmd}} = \mathbf{J}^T\left(\mathbf{\Lambda}\left(\ddot{\mathbf{x}}_{\text{des}} - \dot{\mathbf{J}}\dot{\mathbf{q}} + \mathbf{J}\mathbf{M}^{-1}\mathbf{h}\right) + \mathbf{\Lambda}\mathbf{\Lambda}_{\text{des}}^{-1}(\mathbf{K}_{\text{des}}\widetilde{\mathbf{x}} + \mathbf{D}_{\text{des}}\dot{\widetilde{\mathbf{x}}}) + (\mathbf{\Lambda}\mathbf{\Lambda}_{\text{des}}^{-1} - \mathbf{I})\mathbf{F}_{ext}\right) \tag{2.15}$$

leads to the desired closed-loop behavior as shown in [35]. Typically, the desired inertia is chosen to be identical with the current robot inertia $\mathbf{\Lambda}_{\text{des}} = \mathbf{\Lambda}$, resulting in a simplified control law without inertia shaping

$$\boldsymbol{\tau}_{\text{cmd}} = \mathbf{J}^T \left(\mathbf{\Lambda} \left(\ddot{\mathbf{x}}_{\text{des}} - \dot{\mathbf{J}}\dot{\mathbf{q}} + \mathbf{J}\mathbf{M}^{-1}\mathbf{h} \right) + \mathbf{K}_{\text{des}}\widetilde{\mathbf{x}} + \mathbf{D}_{\text{des}}\dot{\widetilde{\mathbf{x}}} \right). \tag{2.16}$$

Another advantage of this formulation is that the measurement of the external wrench $\mathbf{F}_{ext}$ is avoided. The control law is based only on the joint angle and velocity readings without requiring an additional expensive and potentially noisy force-torque sensor.

Note that realizing a rotational spring in orientation space is not as straightforward as implementing a translational spring [36]. Further note that an accurate dynamic model is assumed here. If this is not the case, any model error will be treated as an external disturbance. The interested reader is referred to [37] for an extensive survey of operational space control.

2.2.2 Prioritizing two Objectives

Robot control in task-space is often considered as more intuitive than control in joint-space because reference trajectories are represented in a low-dimensional task-space in contrast to a representation in the high-dimensional joint-space. For example, planning trajectories while avoiding obstacles tends to be easier and more intuitive in Cartesian space.

Considering a redundant robot there exist infinite sets of joint angle configurations, all satisfying a particular position of an operational point in task-space [11, 15]. Tracking a closed trajectory in task-space, e.g. a circular gesture with the end-effector, does not necessarily lead to a closed trajectory in joint-space, which is different from non-redundant robots with $H = D$. When controlling redundant systems in task-space, the question arises how to select a specific joint angle configuration, also called *redundancy resolution*. This additional complexity is also advantageous at the same time. It offers the possibility to consider more than one control objective simultaneously. Reference torques from various control laws can be superimposed to achieve a range of tasks simultaneously. Prioritization schemes are the preferred means to deal with potentially conflicting objectives. Tasks are often decoupled by projection onto different subspaces.

Considering redundant robots with respect to a Jacobian $\mathbf{J}$, joint velocities $\dot{\mathbf{q}}$ can be separated into two sets:

- The set of all solutions to the equation $\mathbf{J}\dot{\mathbf{q}} \neq \mathbf{0}$ corresponds to the range-space of the Jacobian $\mathbf{J}$. Those joint velocities will affect the motion of the operational point.
- The set of all solutions to the equation $\mathbf{J}\dot{\mathbf{q}} = \mathbf{0}$ corresponds to the nullspace of the Jacobian $\mathbf{J}$. Accordingly, those joint velocities will have *null* effect on the operational point. Instead, those joint velocities result in so-called *eigenmotion* or *self-motion* and affect the redundancy resolution only.

Based on these considerations, orthogonal projectors onto the range $\mathbf{H} \in \mathbb{R}^{D \times D}$ and onto the nullspace $\mathbf{N} \in \mathbb{R}^{D \times D}$ are defined as

$$\mathbf{H} = \mathbf{J}^{+}\mathbf{J} = \mathbf{J}^T \left(\mathbf{J}\mathbf{J}^T \right)^{-1} \mathbf{J} = \mathbf{J}^T (\mathbf{J}^{+})^T \tag{2.17}$$

and

$$\mathbf{N} = \mathbf{I} - \mathbf{H} = \mathbf{I} - \mathbf{J}^{+}\mathbf{J} = \mathbf{I} - \mathbf{J}^T \left(\mathbf{J}\mathbf{J}^T \right)^{-1} \mathbf{J} = \mathbf{I} - \mathbf{J}^T (\mathbf{J}^{+})^T, \tag{2.18}$$

employing the Moore-Penrose inverse. Consequently, Eq. 2.3 and Eq. 2.6 are extended for redundant robots, yielding

$$\dot{\mathbf{q}} = \mathbf{H}\,\mathbf{J}^{+}\dot{\mathbf{x}} + \mathbf{N}\dot{\mathbf{q}}_0 \tag{2.19}$$

and

$$\boldsymbol{\tau} = \mathbf{H}\,\mathbf{J}^T\,\mathbf{F} + \mathbf{N}\boldsymbol{\tau}_0, \tag{2.20}$$

where any arbitrary joint velocity $\dot{\mathbf{q}}_0 \in \mathbb{R}^D$ or torque vector $\boldsymbol{\tau}_0 \in \mathbb{R}^D$ may be projected onto the nullspace (by multiplication with $\mathbf{N}$) and the projection onto the range (by multiplication with $\mathbf{H}$) emphasizes the separation into two different subspaces. Projecting $\mathbf{J}^+$ or $\mathbf{J}^T$ onto its own nullspace cancels itself out and has *null* effect

$$\mathbf{N}\,\mathbf{J}^+ = \mathbf{0} = \mathbf{N}\,\mathbf{J}^T, \tag{2.21}$$

and projection onto its range represents an identity map

$$\mathbf{J}^+ = \mathbf{H}\,\mathbf{J}^+ \text{ and } \mathbf{J}^T = \mathbf{H}\,\mathbf{J}^T. \tag{2.22}$$

Hence, prioritization between a primary (A) and secondary (B) task within velocity-control $\mathbf{J}_\mathrm{A}^+\dot{\mathbf{x}}_\mathrm{A}$, $\dot{\mathbf{q}}_0 = \mathbf{J}_\mathrm{B}^+\dot{\mathbf{x}}_\mathrm{B}$ with Eq. 2.19 or torque-control $\mathbf{J}_\mathrm{A}^T\mathbf{F}_\mathrm{A}$, $\boldsymbol{\tau}_0 = \mathbf{J}_\mathrm{B}^T\mathbf{F}_\mathrm{B}$ with Eq. 2.20 can be imposed by projection onto the orthogonal subspaces $\mathbf{H}$, $\mathbf{N}$ of $\mathbf{J}_\mathrm{A}^+$ and $\mathbf{J}_\mathrm{A}^T$ respectively. With the operational space formulation, task-space velocities $\dot{\mathbf{x}}_\mathrm{A}$, $\dot{\mathbf{x}}_\mathrm{B}$ or task-space wrenches $\mathbf{F}_\mathrm{A}$, $\mathbf{F}_\mathrm{B}$ applied to two different operational points become decoupled in joint-space. Instead of controlling another operational point as secondary task (B), the remaining DOFs in the nullspace can be controlled in joint-space, potentially phrased as a potential function and applying gradient descent. Typical choices are for example staying close to a particular joint-space posture, avoiding obstacles or mechanical joint limits, minimizing energy consumption or optimizing manipulability.

In the following, properties of projection matrices are recalled here: A square projection matrix $\mathbf{X}$ is idempotent by definition, i.e. $\mathbf{X} = \mathbf{X}^2$. It is an orthogonal projection iff $\mathbf{X} = \mathbf{X}^T$, otherwise the projection is referred to as oblique. Two projectors $\mathbf{X}$ and $\mathbf{Y}$ project onto orthogonal subspaces iff $\mathbf{X}^T\,\mathbf{Y} = \mathbf{0}$.

Note that range and nullspace may be complementary subspaces and are not necessarily orthogonal to each other [38, Eq. 5.10.1] and similar holds for projections onto those spaces [38, Eq. 5.9.11]. In robot control often weighted (also referred to as oblique) projection matrices are employed utilizing a weighting matrix $\mathbf{W}$ in the pseudoinverse. The previous projectors onto the range Eq. 2.17 become

$$\mathbf{H}_{\dot{\mathbf{q}}}^{\mathbf{W}} = \quad \mathbf{J}^{\mathbf{W},+}\mathbf{J} \quad = \mathbf{W}\mathbf{J}^T\big(\mathbf{J}\mathbf{W}\mathbf{J}^T\big)^{-1}\mathbf{J} \tag{2.23}$$

$$\mathbf{H}_{\boldsymbol{\tau}}^{\mathbf{W}} = \mathbf{J}^T(\mathbf{J}^{\mathbf{W},+})^T = \mathbf{J}^T\big(\mathbf{J}\mathbf{W}\mathbf{J}^T\big)^{-1}\mathbf{J}\mathbf{W} \tag{2.24}$$

and nullspace projectors $\mathbf{N}_{\dot{\mathbf{q}}}^{\mathbf{W}} = \mathbf{I} - \mathbf{H}_{\dot{\mathbf{q}}}^{\mathbf{W}}$, $\mathbf{N}_{\boldsymbol{\tau}}^{\mathbf{W}} = \mathbf{I} - \mathbf{H}_{\boldsymbol{\tau}}^{\mathbf{W}}$ respectively. The index $()_{\dot{\mathbf{q}}}$ denotes projectors for the weighted Jacobian pseudoinverse $\mathbf{J}^{\mathbf{W},+}$ within velocity-control and index $()_{\boldsymbol{\tau}}$ projectors for the Jacobian transpose $\mathbf{J}^T$ within torque-control. The idempotence criterion is fulfilled even though these projectors are non-orthogonal: $\mathbf{H}^{\mathbf{W}}\mathbf{H}^{\mathbf{W}} = \mathbf{H}^{\mathbf{W}}$, $\mathbf{N}^{\mathbf{W}}\mathbf{N}^{\mathbf{W}} = \mathbf{N}^{\mathbf{W}}$. For oblique projections yields similar as for orthogonal projections that projecting $\mathbf{J}^+$ or $\mathbf{J}^T$ onto its own nullspace cancels itself out and has null effect

$$\mathbf{N}_{\dot{\mathbf{q}}}^{\mathbf{W}}\,\mathbf{J}^+ = \mathbf{0} = \mathbf{N}_{\boldsymbol{\tau}}^{\mathbf{W}}\,\mathbf{J}^T \tag{2.25}$$

and that projection onto its range represents an identity map

$$\mathbf{J}^+ = \mathbf{H}_{\dot{\mathbf{q}}}^{\mathbf{W}}\,\mathbf{J}^+ \text{ and } \mathbf{J}^T = \mathbf{H}_{\boldsymbol{\tau}}^{\mathbf{W}}\,\mathbf{J}^T. \tag{2.26}$$

Weighted projectors can be employed for prioritizing tasks within Eq. 2.19 and Eq. 2.20 as well. Two different properties of consistency are distinguished [15]:

- A nullspace projection is *statically consistent*: The secondary task $\boldsymbol{\tau}_0$ does not generate interfering forces in the task-space of the primary task, i.e. $\mathbf{F} = \mathbf{0}$, in any static equilibrium (steady-state with $\dot{\mathbf{q}} = \ddot{\mathbf{q}} = \mathbf{0}$). This property is fulfilled when projecting $\boldsymbol{\tau}_0$ onto the orthogonal nullspace of the primary task.
- A nullspace projection is *dynamically consistent*: The secondary task $\boldsymbol{\tau}_0$ never generates accelerations in the task-space of the primary task, i.e. $\ddot{\mathbf{x}} = \mathbf{0}$. This property is fulfilled when projecting $\boldsymbol{\tau}_0$ onto the weighted nullspace of the primary task and choosing the inverse of the joint-space inertia as weighting matrix $\mathbf{W} = \mathbf{M}^{-1}$, neglecting the effect of the term $\dot{\mathbf{J}}\,\dot{\mathbf{q}}$.

2.2.3 Prioritizing multiple Objectives

From this point the thesis focuses on torque-control. The index $()_{\boldsymbol{\tau}}$ is dropped for convenience.

When controlling robots with many degrees of freedom such as humanoids, it is a common approach to control multiple prioritized tasks simultaneously. Typical objectives are impedance-based trajectory tracking with feet and hands, orientating a camera located in the head, maintaining the horizontal position of the robot's CoM inside the support polygon, avoiding obstacles and self-collisions, controlling the robot's Centroidal linear/angular momentum, avoiding singularity configurations, damping joint velocities and maintaining a desired body posture. When manipulating a rigid object or an articulated part of the environment additional control objectives are the pose of a manipulated object or the target configuration of an articulated environment part [9]. [10] controls the Center-of-Mass of a group of legged robots, considering all robots together as a single highly redundant multi-body system.

As shown in the previous subsection, the projection onto the nullspace of a redundant task allows to control a secondary task which does not interfere with the primary task. The Stack-of-Tasks (SoT) scheme extends this idea to multi-task prioritization, realizing a *strict hierarchy* of K decoupled operational space tasks. Less important tasks are projected recursively onto the nullspace $\mathbf{N}_i \in \mathbb{R}^{D\times D}$ of higher-priority tasks, thereby ensuring decoupling [13–16]. The final joint torques or joint velocities send to the robot are then computed by

$$\boldsymbol{\tau} = \sum_{i=1}^{K} \mathbf{N}_i\, \boldsymbol{\tau}_i \text{ with } \mathbf{N}_1 = \mathbf{I}, \tag{2.27}$$

where the i-th task is represented by the torque vector $\boldsymbol{\tau}_i$. For hierarchies with more than two tasks, two slightly different nullspace formulations can be distinguished [15]: *successive* and *augmented* projectors $\mathbf{N}_i$. Choosing a strict order of K operational space tasks defined by $\mathbf{J}_1, ..., \mathbf{J}_K$, where $\mathbf{J}_1$ is the Jacobian of the task with highest priority and $\mathbf{J}_K$ with the lowest priority, for all tasks $i > 1$ the successive nullspace projection matrices are obtained by

$$\mathbf{N}_i = \mathbf{N}_{i-1}\left[\mathbf{I} - \mathbf{J}_{i-1}^{T}(\mathbf{J}_{i-1}^{\mathbf{W},+})^{T}\right] \tag{2.28}$$

or, alternatively, augmented nullspace projection matrices by

$$\mathbf{N}_i = \left[\mathbf{I} - \begin{pmatrix}\mathbf{J}_1\\ \vdots \\ \mathbf{J}_{i-1}\end{pmatrix}^{T} \left(\begin{pmatrix}\mathbf{J}_1\\ \vdots \\ \mathbf{J}_{i-1}\end{pmatrix}^{\mathbf{W},+}\right)^{T}\right]. \tag{2.29}$$

The augmented nullspace projections Eq. 2.29 are idempotent and ensure that tasks are projected onto subspaces with dedicated available degrees of freedom (DOFs) implying strict compliance with the task priority order, whereas successive nullspace projections Eq. 2.28 are not idempotent and hence strictly mathematically no projectors but can be computed more efficiently. More theoretical differences between both concepts are highlighted in [15]. Even though

both approaches differ slightly, the evaluation in [15] showed that no significant differences can be observed when applied to a real robot, probably due to inaccuracies of the dynamic model. The SoT scheme can be employed with weighted nullspace projectors similarly. A formal proof of asymptotic stability is provided in [31] for the regulation case.

The idea of SoT has also been formulated as a "cascade" of quadratic programs [39, 40]. Recent approaches in this domain solve the optimization problem with regard to multiple strictly hierarchized equality and inequality constraints at runtime within 1 ms update rate [17–19]. There exist also a number of SoT variants, such as the "Intermediate Desired Value Approach" [41], "Saturation in the Nullspace" [42], or "Generalized Hierarchical Control" [43] to name just a few.

As an alternative approach to strict prioritization within the SoT, a *soft prioritization* between tasks is achieved by assigning scalar priorities [24–26]. Reference torques obtained from a multitude of control objectives are fused employing a weighted sum. The applied net torque results from the linear combination

$$\boldsymbol{\tau} = \sum_{i=1}^{K} \lambda_i \boldsymbol{\tau}_i \ , \tag{2.30}$$

where the generated motion depends on the choice of unit-less scalar priorities $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_K)^T$. Not all controllers have to operate in a low-dimensional task-space as is the case for SoT. Instead, multiple joint-space controllers (with $\mathbf{J} = \mathbf{I} \in \mathbb{R}^{D \times D}$) can be fused easily. In case an exact solution exists with all objectives satisfied, the solution will be found from both soft and strict prioritization schemes. Note that direct superposition of joint torques is semantically meaningful in contrast to joint angles, where superposition can lead to geometrically infeasible solutions. Indeed, psychophysical studies with human subjects support the superposition of torques in ankle angle control for unperturbed and perturbed stance [44]. Note that Eq. 2.30 presents the analytical formulation for soft prioritization, but the same concept can also be employed in quadratic programming [24, 25].

On the one hand, the torque mixture enables to impose soft priorities flexibly and is computationally faster than SoT because it does not rely on projection matrices. On the other hand, it is important to tune the set of priorities $\boldsymbol{\lambda}$ appropriately to achieve a suitable interaction among the different control objectives which is difficult because objectives are not decoupled.

2.2.4 Dealing with Singularities

So far always non-singular configurations were considered with full-rank Jacobian. It is a well-known fact that the classical range and nullspace projections are not suitable for singular configurations, because $\mathbf{JWJ}^T$ drops rank and is not invertible. The "Damped Least Squares" (DLS) technique [20] adds a regularization term $\mathbf{Z}$ turning a singular matrix into a regular one which guarantees full-rank and therefore enables inversion of $\mathbf{JWJ}^T + \mathbf{Z}$. The weighted nullspace projection becomes

$$\mathbf{N}_{\mathrm{DLS}} = \mathbf{I} - \mathbf{J}^T \big(\mathbf{JWJ}^T + \mathbf{Z}\big)^{-1} \mathbf{JW}. \tag{2.31}$$

In the simplest case, $\mathbf{Z} = z^2\ \mathbf{I}$ with constant $z \in \mathbb{R}$. Unfortunately, $\mathbf{JWJ}^T$ regularized by $\mathbf{Z}$ leads to a sub-optimal solution. When applying DLS regularization, $\mathbf{N}_{\mathrm{DLS}}$ violates the idempotence criterion but can be seen as an approximation: $\mathbf{N}_{\mathrm{DLS}}^2 \approx \mathbf{N}_{\mathrm{DLS}}$. The resulting matrix operator $\mathbf{N}_{\mathrm{DLS}}$ is not a projector onto the nullspace of $\mathbf{J}^T$ any longer, the null-effect property is violated: $\mathbf{0} \neq \mathbf{N}_{\mathrm{DLS}}\mathbf{J}^T$. Hence, two strictly prioritized operational space tasks are not fully decoupled. When employing SoT with regularized nullspace projections Eq. 2.31, errors resulting from the regularization term will propagate to the last level. Choosing a damping $\mathbf{Z}$ is a compromise between robustness in singular configurations and accuracy. This issue is tackled by approaches

with variable damping [45]. However, choosing $\mathbf{Z}$ according to these approaches is not intuitive because of additional meta-parameters and its direct consequences are not obvious. In [18], each task in a SoT is regularized individually.

2.3 Multi-Objective Robot Control in Contact Situation

Robot control in multi-contact situation introduces an additional objective besides generating constraint-consistent motion: realizing desired contact wrenches. Maintaining the contact constraint is a challenging research field. Decoupling these two main objectives allows two employ different and independent the control laws. Throughout this thesis a rigid contact situation is assumed to simplify matters. Assuming bilateral contact constraints with zero Cartesian velocities and accelerations at the contact points the constraint is modeled as

$$\mathbf{J}_c\,\dot{\mathbf{q}} = \mathbf{0} \;\text{ and }\; \mathbf{J}_c\,\ddot{\mathbf{q}} + \dot{\mathbf{J}}_c\,\dot{\mathbf{q}} = \mathbf{0}\;, \tag{2.32}$$

where $\mathbf{J}_c$ describes the constrained Jacobian associated with the contact situation, e.g. a vertical concatenation of all Jacobians associated with contact points. Unilateral contact constraints may be modeled with Eq. 2.32 as well when guaranteeing that the generated contact wrenches $\mathbf{F}_c$ maintain the contact. Contact force and contact moment must satisfy additional constraints imposed by contact friction.

2.3.1 Contact Constraints in Grasping Situations

Consider in the following a robot system with a total of D joints composed of $B > 1$ manipulators manipulating a rigid object via a rigid grasp with all end-effectors in contact. Hence, end-effectors are *virtually linked* to each other [46], i.e. the relative transformation between them stays constant, representing a closed kinematic chain (simple case) or a closed kinematic tree structure (general case).

Following the definition in [47], the *partial grasp matrix* respect to a global coordinate frame $\mathbf{G}_i \in \mathbb{R}^{6\times 6}$ of the i-th arm in a multi-arm system is defined by the mapping between the object twist to the contact twists [36]:

$$\mathbf{G}_i = \begin{bmatrix} \mathcal{R}_i & \mathbf{0} \\ \mathbb{S}(\boldsymbol{r}_i) & \mathcal{R}_i \end{bmatrix}, \tag{2.33}$$

where $\mathcal{R}_i$ represents the rotation matrix of the i-th contact frame, $\boldsymbol{r}_i$ is the distance between the i-th contact position to the object Center-of-Mass (i.e. the so-called virtual stick [48]), and $\mathbb{S}(\boldsymbol{r}) \in \mathbb{R}^{3\times 3}$ is the skew-symmetric matrix that performs the cross product

$$\mathbb{S}(\boldsymbol{r}) = \begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix}. \tag{2.34}$$

The *complete grasp matrix* $\mathbf{G} \in \mathbb{R}^{6\times 6\,B}$ is the horizontal concatenation of all B partial grasp matrices, representing the relative transformations between all end-effectors

$$\mathbf{G} = [\mathbf{G}_1, ..., \mathbf{G}_B]\,. \tag{2.35}$$

The projector $\mathbf{I} - \mathbf{G}^T\big(\mathbf{G}\,\mathbf{G}^T\big)^{-1}\mathbf{G} \in \mathbb{R}^{6\,B\times 6\,B}$ projects any arbitrary vector onto the nullspace of the grasp matrix. The resulting contact wrench is usually referred as the internal wrench, since it produces no net wrench, i.e., $\mathbf{G}\mathbf{F}_c = \mathbf{0}$. During grasping, the constraints should enforce the rigid grasp without compromising motion generation. Contact wrenches are controlled such that only the internal wrench is allowed [49] [50]. For this reason, the constraint Jacobian is

$$\mathbf{J}_c \in \mathbb{R}^{6B\times D} = \Big(\mathbf{I} - \mathbf{G}^T\big(\mathbf{G}\,\mathbf{G}^T\big)^{-1}\mathbf{G}\Big)\,\mathbf{J}_{ee}\;, \tag{2.36}$$

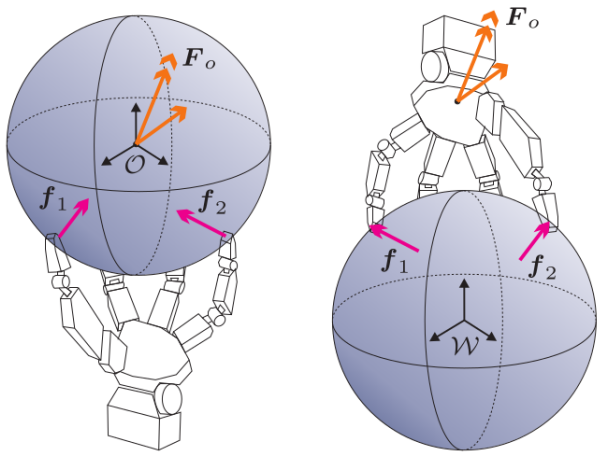


Figure 2.2: The wrench distribution is fundamentally similar in multi-arm grasping (left) and multi-leg balancing (right). The image is taken from [36].

where $\mathbf{J}_{ee}$ denotes the block-diagonal combination of each i-th manipulator Jacobian

$$\mathbf{J}_{ee} = \begin{bmatrix} \mathbf{J}_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{J}_B \end{bmatrix}. \tag{2.37}$$

Note that $\mathbf{J}_c$ constrains only the relative motion between manipulators. The constraint itself is allowed to move. Further note that $\mathbf{J}_c$ is rank deficient. Since all manipulators are independent from each other, the inertia matrix $\mathbf{M} \in \mathbb{R}^{D \times D}$ of the multi-arm robot system is block-diagonal

$$\mathbf{M} = \begin{bmatrix} \mathbf{M}_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{M}_B \end{bmatrix}, \tag{2.38}$$

and $\mathbf{h} \in \mathbb{R}^D$ is the vertical concatenation of each individual manipulators $\mathbf{h}_i$

$$\mathbf{h} = \left[\mathbf{h}_1^T, ..., \mathbf{h}_B^T\right]^T. \tag{2.39}$$

It has been recognized that multi-contact situations with multi-leg robots are fundamentally similar to grasping situations [36, 51, 52]. Fig. 2.2 portraits both scenarios: the robot tries to achieve a desired wrench via the contact points while the relative transformation between contacts stays constant. Hence, the grasp matrix originally proposed for multi-fingered hands can also be employed for legged robots in multi-contact situations.

2.3.2 Stack-of-Tasks in Contact Situation

The Stack-of-Tasks scheme can be employed to deal with contact constraints [53]. Contact wrenches $\mathbf{F}_c$ are generated by the *constrained controller*

$$\boldsymbol{\tau}_c = \mathbf{J}_c^T \mathbf{F}_c \,. \tag{2.40}$$

Typically, the z-axis is chosen as the direction normal to the contact surface. Contact wrench control is considered in the primary level of a strict nullspace hierarchy to guarantee the constraints. Motion tasks are projected onto the nullspace of the constraints, therefore ensuring that robot motion does not violate the contact constraint (i.e. the motion controllers contribute no acceleration at the constraint locations). The control scheme is designed to remain dynamically

consistent with respect to the contact constraints by employing the inertia-weighted nullspace projection $\mathbf{N}_c$ of the constraint. The final torques $\boldsymbol{\tau}_{\text{cmd}}$ send to the robot become

$$\boldsymbol{\tau}_{\text{cmd}} = \boldsymbol{\tau}_c + \mathbf{N}_c \boldsymbol{\tau}_u \ , \tag{2.41}$$

where the *unconstrained controller* $\boldsymbol{\tau}_u \in \mathbb{R}^D$ performs motion generation in joint-space or task-space or may consider multiple prioritized objectives.

2.3.3 Projected Inverse Dynamics Control (PIDC)

Contact constraints, by definition, are indeed already dynamic consistent, because they do not generate acceleration. Consequently, constraints are able to apply necessary wrenches to maintain their own consistency, and control schemes do not need to address the dynamic consistency property explicitly.

Accordingly, the Projected Inverse Dynamics Control (PIDC) approach [27] employs an orthogonal nullspace $\mathbf{N}_c$ only without imposing a specific weighting. This is advantageous, because accurately determining inertial parameters [37] is difficult and a orthogonal projection is derived only from kinematic parameters. Also, real-robot experiments proved that incorporating the inertia matrix in practice does not significantly show the theoretical conceptual superiority. The authors in [15] realize only minor improvements in tracking performance.

Within PIDC, motion control $\boldsymbol{\tau}_u$ may be performed in joint-space ($\boldsymbol{\tau}_u = \mathbf{M}\ddot{\mathbf{q}} + \mathbf{h}$) as shown in [27], or in operational space ($\boldsymbol{\tau}_u = \mathbf{J}^T\mathbf{F}$) as proposed in [28], or may consider multiple prioritized objectives as demonstrated in [54]. In [54] the approach was validated with a torque-controlled manipulator for wiping a board.

Joint-Space Control with PIDC

In contact situation, any admissible joint velocity must belong to the nullspace of the constraint Jacobian matrix $\mathbf{J}_c$, thus

$$\mathbf{H}_c\,\dot{\mathbf{q}} = \mathbf{0} \text{ and } \mathbf{N}_c\,\dot{\mathbf{q}} = \dot{\mathbf{q}} \ , \tag{2.42}$$

and time differentiation yields

$$\mathbf{N}_c\,\ddot{\mathbf{q}} + \dot{\mathbf{N}}_c\,\dot{\mathbf{q}} = \ddot{\mathbf{q}} \text{ or equivalently } (\mathbf{I} - \mathbf{N}_c)\ddot{\mathbf{q}} = \dot{\mathbf{N}}_c\,\dot{\mathbf{q}} \ . \tag{2.43}$$

Next decomposing the rigid body dynamics Eq. 2.9 onto two orthogonal subspaces by left-multiplication with $\mathbf{N}_c$ (motion subspace or unconstrained space) and $\mathbf{H}_c = \mathbf{I} - \mathbf{N}_c$ (wrench subspace or constrained space) yields

$$\mathbf{N}_c\boldsymbol{\tau}_u = \mathbf{N}_c\left(\mathbf{M}\ddot{\mathbf{q}} + \mathbf{h} + \mathbf{J}_c^T\mathbf{F}_c\right) = \mathbf{N}_c\left(\mathbf{M}\ddot{\mathbf{q}} + \mathbf{h}\right) \tag{2.44}$$

and

$$(\mathbf{I} - \mathbf{N}_c)\boldsymbol{\tau}_c = (\mathbf{I} - \mathbf{N}_c)\left(\mathbf{M}\ddot{\mathbf{q}} + \mathbf{h} + \mathbf{J}_c^T\mathbf{F}_c\right). \tag{2.45}$$

Adding Eq. 2.43 and Eq. 2.44 results in

$$\begin{aligned} \mathbf{N}_c\,\mathbf{M}\ddot{\mathbf{q}} + (\mathbf{I} - \mathbf{N}_c)\ddot{\mathbf{q}} &= \mathbf{N}_c\boldsymbol{\tau}_u - \mathbf{N}_c\,\mathbf{h} + \dot{\mathbf{N}}_c\,\dot{\mathbf{q}} \\ \mathbf{M}_c\,\ddot{\mathbf{q}} &= \mathbf{N}_c\boldsymbol{\tau}_u - \mathbf{N}_c\,\mathbf{h} + \dot{\mathbf{N}}_c\,\dot{\mathbf{q}} \end{aligned} \tag{2.46}$$

with the constrained inertia matrix $\mathbf{M}_c = \mathbf{N}_c\mathbf{M} + \mathbf{I} - \mathbf{N}_c$ which is invertible. Eq. 2.46 multiplied with $\mathbf{M}_c^{-1}$ gives the equation of motion

$$\begin{aligned} \ddot{\mathbf{q}} &= \mathbf{M}_c^{-1}\mathbf{N}_c\boldsymbol{\tau}_u - \mathbf{M}_c^{-1}\mathbf{N}_c\mathbf{h} + \mathbf{M}_c^{-1}\dot{\mathbf{N}}_c\,\dot{\mathbf{q}} \\ \ddot{\mathbf{q}} &= \mathbf{M}_c^{-1}(\mathbf{N}_c(\boldsymbol{\tau}_u - \mathbf{h}) + \dot{\mathbf{N}}_c\,\dot{\mathbf{q}}). \end{aligned} \tag{2.47}$$

Note that joint accelerations are not affected by contact wrenches. Or equivalently, contact wrenches do not generate robot motion. Substituting $\ddot{\mathbf{q}}$ from Eq. 2.47 into Eq. 2.45 results in

$$\begin{aligned}(\mathbf{I}-\mathbf{N}_c)\boldsymbol{\tau}_c &= (\mathbf{I}-\mathbf{N}_c)\left(\mathbf{M}\ddot{\mathbf{q}}+\mathbf{h}+\mathbf{J}_c^T\mathbf{F}_c\right)\\ (\mathbf{I}-\mathbf{N}_c)\boldsymbol{\tau}_c &= (\mathbf{I}-\mathbf{N}_c)\left(\mathbf{M}\mathbf{M}_c^{-1}(\mathbf{N}_c(\boldsymbol{\tau}_u-\mathbf{h})+\dot{\mathbf{N}}_c\,\dot{\mathbf{q}})+\mathbf{h}+\mathbf{J}_c^T\mathbf{F}_c\right)\\ (\mathbf{I}-\mathbf{N}_c)\boldsymbol{\tau}_c &= (\mathbf{I}-\mathbf{N}_c)\left(\mathbf{J}_c^T\mathbf{F}_c+\mathbf{h}+\boldsymbol{\epsilon}\right), \end{aligned} \tag{2.48}$$

with $\boldsymbol{\epsilon}=\mathbf{M}\mathbf{M}_c^{-1}(\mathbf{N}_c(\boldsymbol{\tau}_u-\mathbf{h})+\dot{\mathbf{N}}_c\,\dot{\mathbf{q}})\in\mathbb{R}^D$ enabling constraint-consistent control even though no dynamically consistent nullspace projection is employed.

The total torque command $\boldsymbol{\tau}_{\text{cmd}}$ applied to the robot is the sum of unconstrained (motion) and constrained (wrench) subspace:

$$\begin{aligned}\boldsymbol{\tau}_{\text{cmd}} &= \mathbf{N}_c\boldsymbol{\tau}_u+(\mathbf{I}-\mathbf{N}_c)\,\boldsymbol{\tau}_c\\ &= \mathbf{N}_c\,(\mathbf{M}\ddot{\mathbf{q}}+\mathbf{h})+(\mathbf{I}-\mathbf{N}_c)\left(\mathbf{J}_c^T\mathbf{F}_c+\mathbf{h}+\boldsymbol{\epsilon}\right). \end{aligned} \tag{2.49}$$

Task-Space Control with PIDC

Within PIDC motion control $\boldsymbol{\tau}_u$ may be performed in an operational space [28] associated with a Jacobian $\mathbf{J}$ that is different from $\mathbf{J}_c$. Therefore, Eq. 2.46 is left-multiplied by $\mathbf{J}\mathbf{M}_c^{-1}$, replacing $\mathbf{J}\mathbf{M}_c^{-1}\mathbf{M}_c\ddot{\mathbf{q}}=\mathbf{J}\ddot{\mathbf{q}}=\ddot{\mathbf{x}}-\dot{\mathbf{J}}\,\dot{\mathbf{q}}$

$$\begin{aligned}\mathbf{J}\mathbf{M}_c^{-1}\mathbf{M}_c\ddot{\mathbf{q}} &= \mathbf{J}\mathbf{M}_c^{-1}\left(\mathbf{N}_c\boldsymbol{\tau}_u-\mathbf{N}_c\mathbf{h}+\dot{\mathbf{N}}_c\,\dot{\mathbf{q}}\right)\\ \ddot{\mathbf{x}}-\dot{\mathbf{J}}\,\dot{\mathbf{q}} &= \mathbf{J}\mathbf{M}_c^{-1}\left(\mathbf{N}_c\boldsymbol{\tau}_u-\mathbf{N}_c\mathbf{h}+\dot{\mathbf{N}}_c\,\dot{\mathbf{q}}\right)\\ \mathbf{J}\mathbf{M}_c^{-1}\mathbf{N}_c\boldsymbol{\tau}_u &= \ddot{\mathbf{x}}-\dot{\mathbf{J}}\,\dot{\mathbf{q}}+\mathbf{J}\mathbf{M}_c^{-1}\left(\mathbf{N}_c\mathbf{h}-\dot{\mathbf{N}}_c\,\dot{\mathbf{q}}\right). \end{aligned} \tag{2.50}$$

Next, left-multiplying with $\mathbf{J}^T\boldsymbol{\Lambda}_c$ and choosing the constraint task-space inertia matrix defined as $\boldsymbol{\Lambda}_c=(\mathbf{J}\mathbf{M}_c^{-1}\mathbf{N}_c\mathbf{J}^T)^{-1}$ results into

$$\boldsymbol{\tau}_u=\mathbf{J}^T\boldsymbol{\Lambda}_c\,\mathbf{J}\mathbf{M}_c^{-1}\mathbf{N}_c\boldsymbol{\tau}_u=\mathbf{J}^T\boldsymbol{\Lambda}_c\left(\ddot{\mathbf{x}}-\dot{\mathbf{J}}\,\dot{\mathbf{q}}+\mathbf{J}\mathbf{M}_c^{-1}\left(\mathbf{N}_c\mathbf{h}-\dot{\mathbf{N}}_c\,\dot{\mathbf{q}}\right)\right). \tag{2.51}$$

Note that $\mathbf{J}^T\boldsymbol{\Lambda}_c\,\mathbf{J}\mathbf{M}_c^{-1}\mathbf{N}_c=\mathbf{J}^T(\mathbf{J}\mathbf{M}_c^{-1}\mathbf{N}_c\mathbf{J}^T)^{-1}\mathbf{J}\mathbf{M}_c^{-1}\mathbf{N}_c$ is a projector onto the range of $\mathbf{J}^T$ with weighting $\mathbf{M}_c^{-1}\mathbf{N}_c$. In case of remaining redundancy, it is also possible to consider multiple prioritized motion objectives by adding a dynamically consistent nullspace with weighting matrix $\mathbf{W}_c=\mathbf{M}_c^{-1}\mathbf{N}_c$.

Chapter 3

Learning Soft Priorities

The soft prioritization scheme does not rely on nullspace projection matrices and accordingly objectives interfere with each other. It is crucial to find a suitable set of priorities to achieve an efficient interaction between all control objectives. To address this issue, this chapter proposes a framework for automated learning of soft priorities. Priority sets are evaluated in simulation and optimized with respect to a cost function. The framework is first evaluated on a planar manipulator as an illustrative example and next on the humanoid robot COMAN for two different high-level goals. Finally, the main advantage of the soft prioritization approach is highlighted: priorities for the underlying control objectives can easily be adapted during motion execution online. In case of changes in the environment or adaptation of the high-level goal, the robot reacts by choosing a new set of priorities according to the previous optimization experiments. The new set is smoothly blended, enabling continuous task-priority rearrangement and resulting in continuous robot motion. Note that large parts of this chapter have already been published in [55] and [56]. The chapter concludes with a detailed comparison of the proposed approach with state-of-the-art before public release and also reviews recent works which further extend the idea of learning priorities.

3.1 Optimization of Soft Priorities

For generation of proper behavior with the soft prioritization scheme, it is crucial to find a set of suitable scalar priorities $\boldsymbol{\lambda}$. This constitutes a high-dimensional search problem of dimension $K = dim(\boldsymbol{\lambda})$. This chapter treats the priorities $\boldsymbol{\lambda}$ as parameters of a policy, which computes for each generalized robot state an action (torques) and transforms the current state into a consecutive state through motion generation. Optimization is based on averaging weighted costs, i.e. random perturbations of the policy parameters $\boldsymbol{\lambda}$ are evaluated in so-called "roll-outs", which execute the respective behavior and compute costs. Stochastic policy search leads to different priorities depending on the cost function, which again has to be designed according to the desired overall high-level goal.

3.1.1 Covariance Matrix Adaptation Evolution Strategy

The state-of-the-art Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [57, 58] is employed for policy improvement w.r.t. a cost function. CMA-ES generates roll-outs $\boldsymbol{\lambda}_k^{(e+1)}$ in episode $e+1$ by sampling from a multivariate normal distribution

$$\boldsymbol{\lambda}_k^{(e+1)} = \boldsymbol{\lambda}^{(e)} + \sigma^{(e)} \mathcal{N}\left(0, \mathbf{C}^{(e)}\right), \tag{3.1}$$

where $k = 1, ..., 4 + \lfloor 3\,log(K) \rfloor$ is the index of the specific roll-out sample, $\boldsymbol{\lambda}^{(e)}$ is the mean value of the search distribution, $\mathbf{C}^{(e)} \in \mathbb{R}^{K \times K}$ is the covariance matrix representing pairwise dependencies between the policy parameters, and $\sigma^{(e)} \in \mathbb{R}^{+}$ is the step-size (or "overall" standard deviation). Mean, covariance matrix and step-size are updated in each episode e as a cost-weighted sum of the best performing policy parameters. This ensures that roll-outs are sampled in the most promising directions of the policy space. CMA-ES converges towards a local optimum. It has been successfully applied for many robotic applications. Costs are summed from multiple criteria, neglecting different units.

The experiments belonging to this chapter employ CMA-ES available as Matlab source code [59], configured as follows: Non-negative priorities are enforced by imposing a lower bound. The "linear decrease" weighting strategy is chosen. Simulating the roll-outs is done without considering actuation noise.

3.1.2 Simulation of Roll-outs

This chapter refers to a roll-out of a specific priority set $\boldsymbol{\lambda}$ as a combination of the initial robot configuration (joint angles, velocities and accelerations) and possibly a Cartesian reaching target, or specific external forces applied to the robot or obstacles positioned in the workspace. A typical problem of optimization, which occurred also in preliminary experiments with the described framework frequently, is "over-fitting". That is, the policy works well for a particular goal (e.g. a specific initial joint configuration) that is tested in the roll-outs, but does not even generalize to very similar goals. To avoid such over-fitting and to learn versatile policies, in this chapter it is proposed to evaluate in each roll-out a set of T representative predefined goals. Thereby all roll-outs can be compared on the same set of goals, where each goal constitutes a separate simulation. All described experiments are performed through a dynamics simulation and refer to the total number of goal simulations in a single CMA-ES update cycle as an episode.

3.1.3 Goal Scores and Cost Function

In all optimization approaches based on cost functions, the choice of the cost (or, equivalently, negative of a reward to be optimized) is highly important. This chapter devises cost functions for different high-level goals.

Intuitively, zero coefficients are preferred for irrelevant control objectives to prevent them from interfering with the relevant ones and to reduce energy consumption. Because CMA-ES does not detect irrelevant policy parameters explicitly, in the following, it is investigated whether it is helpful to add a regularization term to the cost function. Since the priorities vary a lot in their scales, it is not possible to simply punish large coefficients or reward small coefficients. Instead, the weighted absolute torques computed by the controllers are summed and normalized by the number of joints D to act as regularization term. This ensures that energy efficient movements are rewarded, and behaviors are punished, where controllers cancel each other out.

In general, the cost $c_k^{(e)}$ for a single roll-out k in episode e is based on the median of the scores $s_1, \ldots, s_T$ for each separate goal simulation. In order to account also for goal simulations with failures (e.g. violated joint limit constraints) a second element is added to the cost which analyzes the standard deviation of all goal scores:

$$c_k^{(e)} = median(s_1, \ldots, s_T) - 0.05\, std(s_1, \ldots, s_T)\ . \tag{3.2}$$

3.1.4 Generalization Test and Improvement Measure

Once the optimization process is stopped, the best of all rolled out priorities $\boldsymbol{\lambda}^*$ is determined. A generalization test is performed to investigate whether this best set of priorities $\boldsymbol{\lambda}^*$ is transferable to new goals. It is evaluated on a new set of goals, and a cost c^* is calculated by the same procedure as described in Sec. 3.1.3.

The cost c_{init} of the initial policy is evaluated on the same set of new test goals. Then, the improvement measure is obtained based on the comparison of both results. Since all costs are positive, the improvement measure is defined as $\frac{c_{init}}{c^*}$.

3.2 Learning Soft Priorities for Reaching Targets

This chapter adopts and partially extends the controllers proposed in [26] for multi-objective motion generation. Besides one objective for respecting *Joint Angle Limits* (JAL), four controllers are employed to maintain balance: *Minimum Effort* (ME), *Joint Momentum* (JM), *Linear and Angular Momenta about Center-of-Mass* (LM and AM). The *End-Effector* (EE) tracks a target in Cartesian Space. Finally, *Gravity Compensation* (GC) in joint-space is added with weight equal to one. Some of the controllers are split into a proportional (P) and derivative (D) part. For more information, the interested reader is referred to [26], where these control objectives were proposed and tested for isolated joints only.

In addition to [26], a controller *Joint Velocity Limit* (JVL) keeps the joint velocities $\dot{\mathbf{q}}$ in an acceptable range, which is active and produces a repelling torque if either $\dot{q}_i^- < \dot{q}_i < \dot{\theta}_i^-$ or $\dot{\theta}_i^+ < \dot{q}_i < \dot{q}_i^+$ holds for a joint i. That is, if the velocity is too close to the velocity limit $\dot{q}_i^+$ and $\dot{q}_j^-$ according to thresholds $\dot{\theta}_i^+$ and $\dot{\theta}_i^-$ respectively. When approaching the lower limit, i.e. $\dot{q}_i < \dot{\theta}_i^-$, the computed repelling torque is

$$\tau_{JVL,i} = \left(\frac{\dot{q}_i - \dot{\theta}_i^-}{\dot{q}_i^- - \dot{\theta}_i^-} \right)^2 . \tag{3.3}$$

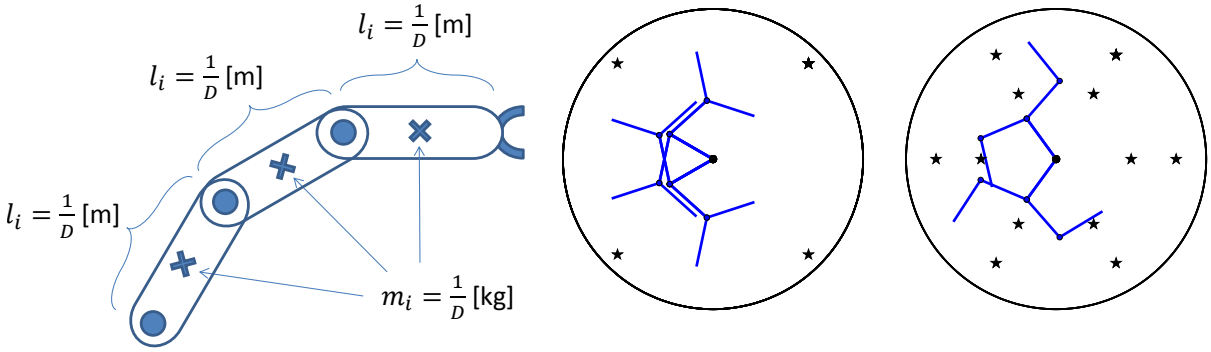
A similar equation holds for approaching the upper limit, while the generated torque of is zero if $\dot{\theta}_i^- < \dot{q}_i < \dot{\theta}_i^+$.

Note that the optimization process learns weights for all control objectives except for the gravity compensation controller which must always equal 1.0 to ensure proper gravity compensation.

As a proof-of-concept and illustrative example, where results are nicely interpretable, the previously described optimization framework is tested first with a simple robot. Fig. 3.1a illustrates a planar virtual manipulator with $D = 3$ revolute joints that is modeled and simulated with Robotran [60]. The desired high-level goal is to reach for targets, starting from arbitrary configurations while respecting joint limits. The total mass of the manipulator is $m = 1$ kg and the total length is $l = 1$ m. Each link has its Center-of-Mass in the midpoint of the link, weighs $m = 1/D$ kg and is $l = 1/D$ m long. Joint velocity limits are set to 2 rad/s and joint angle limits to $\pm\pi$ for all joints except the first one which can oscillate around 360 degrees. Gravity is the only external force acting upon the robot.

Reaching targets are generated which are equally distributed in the reachable workspace, i.e. four reaching targets within the circle of radius 1 m around the base of the manipulator. Initial conditions comprise two different joint angle displacements for each joint, resulting in a set of $2^3 \cdot 4 = 32$ different evaluations per roll-out. The manipulator starts at rest, i.e. initial joint velocities $\dot{\mathbf{q}} = \mathbf{0}$ and accelerations $\ddot{\mathbf{q}} = \mathbf{0}$. The test targets are generated similarly. Fig. 3.1 shows the considered reaching targets, where symmetry was exploited to reduce the number of targets.

To obtain the cost, a goal performance score s is computed first for each goal performance as the combination of three criteria. Considering the high-level goal of reaching for target points, a natural criterion is the achieved distance d_{end} between the end-effector and the target. The duration t_{end} until the robot is at rest and the overall energy consumption b are added. The latter is computed by integrating the torques applied to the joints during simulation and normalized by the number of joints to allow comparison between different robot models. The



(a) Simulated manipulator with $D = 3$ joints (b) 32 training goals (c) 48 test goals

Figure 3.1: Robot model (a) as well as training (b) and test (c) goal sets for the 3 DOF manipulator.

goal score s is then computed as

$$s = \underbrace{-\alpha d_{\text{end}}}_{\text{primary}} \underbrace{-\beta t_{\text{end}} - \gamma b}_{\text{secondary}}, \tag{3.4}$$

where scale factors $\alpha = 15.0$, $\beta = 1.0$ and $\gamma = 0.005$ express that the primary criterion is target-reaching and more important than the secondary criteria of minimum duration and energy efficiency. This is crucial because otherwise it can happen that an undesirable policy is learned, which does not move at all, i.e. has a minimum of time till rest and negligible energy costs, but never reaches a target. Regularization to detect irrelevant control objectives is not tackled in this example.

The score is computed if the robot reaches the target. Two other cases are handled separately. First, if after a duration of maximum four seconds reaching has not converged, the simulation is stopped and d_{end} is set to the maximum possible distance. Second, if the generated movement exceeds joint angle or velocity limits, the simulation is stopped prematurely and the worst possible goal score is assigned to this simulation based on the maximum possible distance $d_{\max}$, duration $t_{\max}$ and energy consumption $b_{\max}$.

Manually tuned initial values for $\boldsymbol{\lambda}_{init}$ according to Tab. 3.1a are used to speed up convergence. The initial standard deviation of CMA-ES is treated as a vector related to the initial policy parameters with $0.2\,\boldsymbol{\lambda}_{init}$.

Table 3.1: Soft priorities (a) and improvement measures in the generalization test (b) for the 3 DOF manipulator.

Coefficient	Initial	Best
λ_{GC}	1.0	1.0
λ_{ME}	0.2	1.2
λ_{JM}	0.5	5.7
λ_{LM}	0.5	0.6
λ_{AM}	1.5	3.0
λ_{JALP}	10.0	68.9
λ_{JALD}	0.5	1.7
λ_{JVL}	15.0	94.2
λ_{KEP}	30.0	177.4
λ_{KED}	5.0	33.6

(a)

Trial	Improvement
1	1.19
2	1.30
3	1.14
4	1.32
5	1.21
6	1.21
7	1.18
8	1.26
9	1.26
10	1.21

(b)

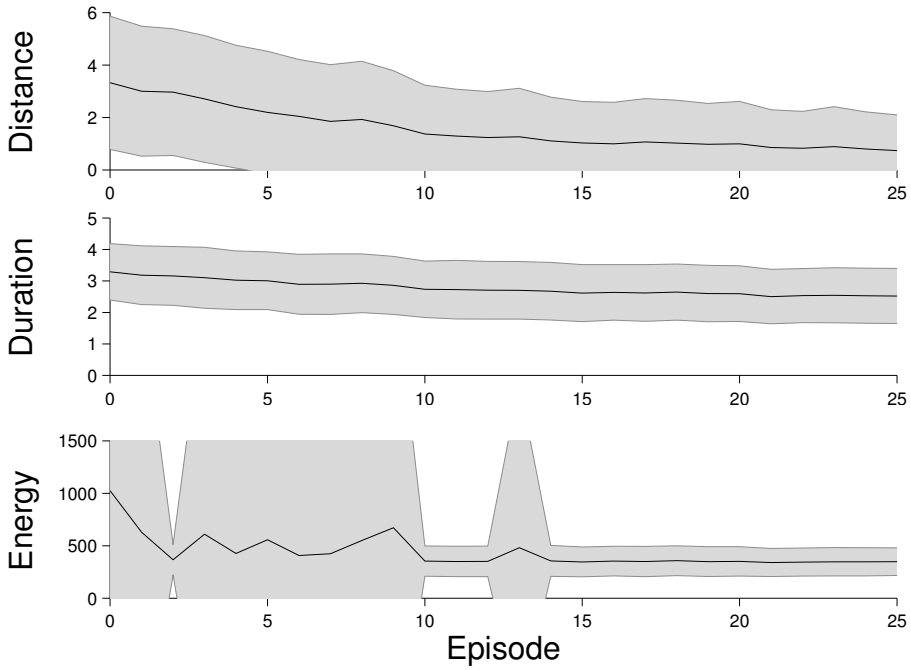


Figure 3.2: Results for the 3 DOF manipulator reaching to targets: The criteria distance, duration and energy consumption of the best roll-out per episode are shown. Mean and standard deviation for all ten trials are plotted.

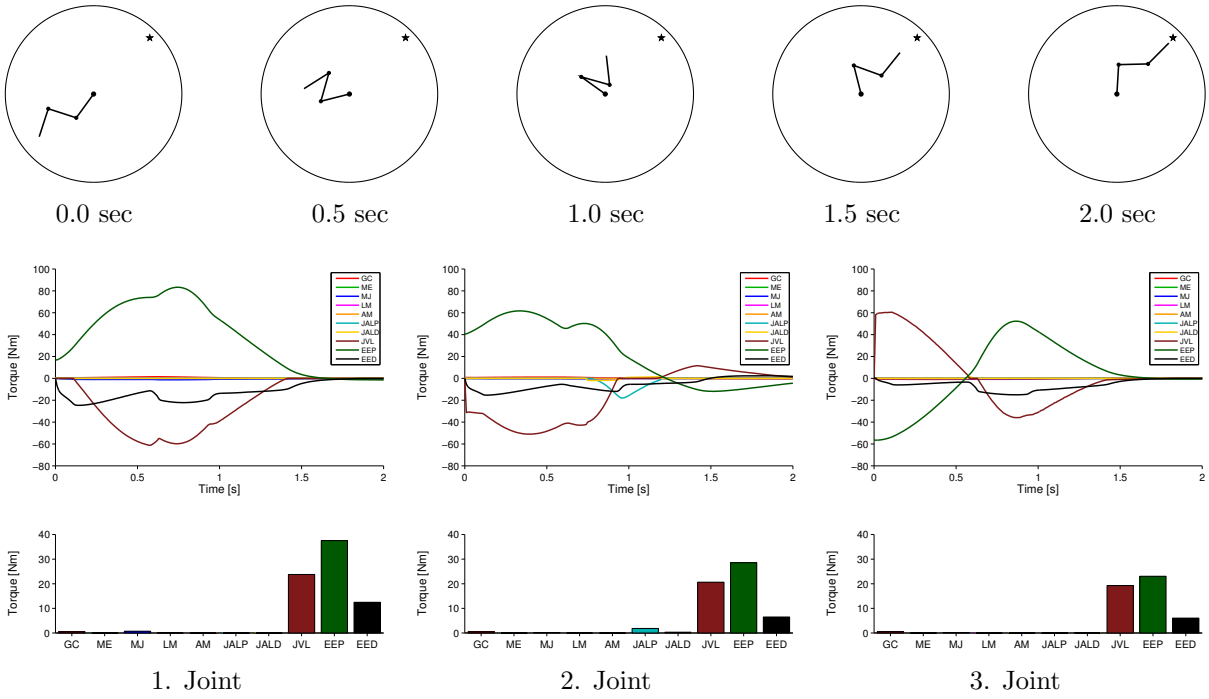


Figure 3.3: Exemplary trajectory of the 3 DOF manipulator with learned priorities (Tab. 3.1a). The top row shows snapshots of the trajectory. The corresponding control torques per time step and joint for each objective are shown in the middle row. The bottom row shows the average of absolute torques over the entire trajectory for each control objective and joint.

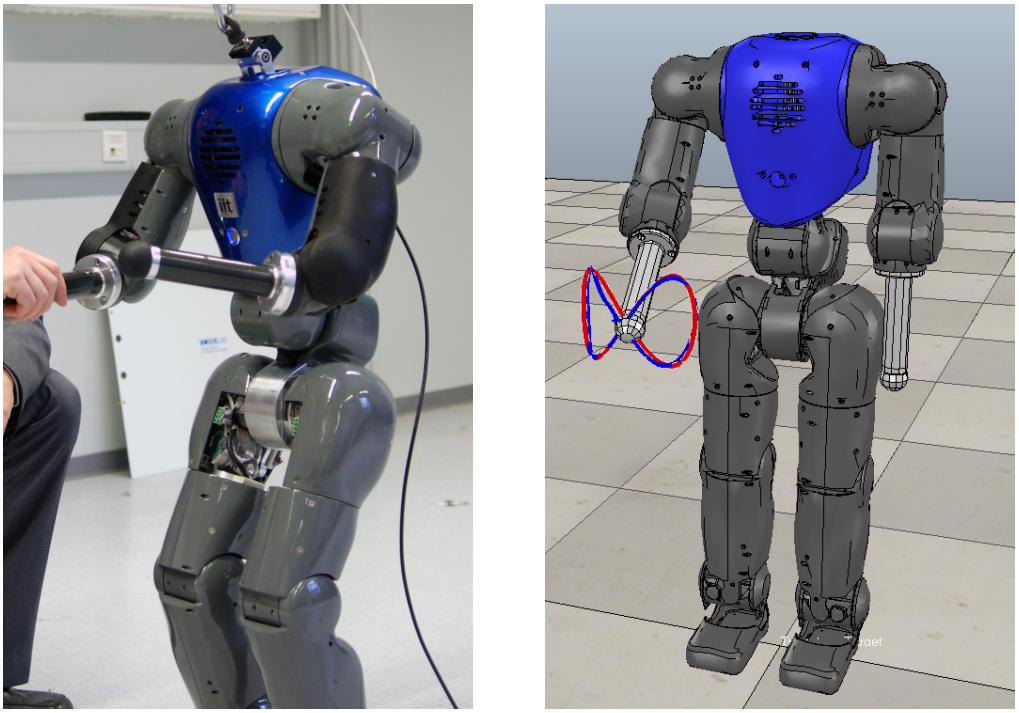


Figure 3.4: Left: The COMpliant huMANoid COMAN [61] is perturbed by a human. Right: Dynamics simulation with end-effector trajectory (blue) and desired trajectory (red). The robot is perturbed while tracking and adapts the priorities to compensate the external forces.

To investigate the reliability of the overall learning framework, the experiment is repeated ten times with different random generator initialization showing consistent behavior in Fig. 3.2. During the first ten episodes, the costs for the best roll-out in each episode decrease fast and then slowly converge. This is as expected because the cost function is bounded from below (cost $c = 0$) and it becomes increasingly difficult to improve costs. In more detail, all three goal score criteria shown in Fig. 3.2 (end-effector target distance d_{end}, duration t_{end} and energy consumption b) consistently decrease during the first episodes and then converge.

The optimization processes are stopped after 25 episodes and the priorities belonging to the overall best roll-out are given in Tab. 3.1a. Tab. 3.1b lists the improvement factors obtained from the generalization test performed with 48 new goals, which are shown in Fig. 3.1c. Generally, the learned policies are transferable to new situations and outperform the initial policy.

Fig. 3.3 shows an exemplary trajectory, which is generated with the learned priorities of the best trial (Tab. 3.1a) and the torques per joint computed by the different controllers. The manipulator starts in a stretched joint angle configuration and approaches steady-state with the end-effector close to the desired reaching target. The complete movement is quick and energy efficient. There are three phases observable, which result in quite different torque profiles. First, the manipulator accelerates and folds up while respecting its joint limits. Next, it rotates compactly around its base. In the last phase, the manipulator slowly extends to reach the target. Fig. 3.3 also shows the torques computed to achieve the different control objectives per joint. As expected, the primary criterion of reaching, as expressed by the EE control objective, is predominant and generates the largest torques. The optimization process increased the corresponding priorities to fulfill this criterion more precisely. Additional JAL and JVL are important to prevent violations of joint limit constraints.

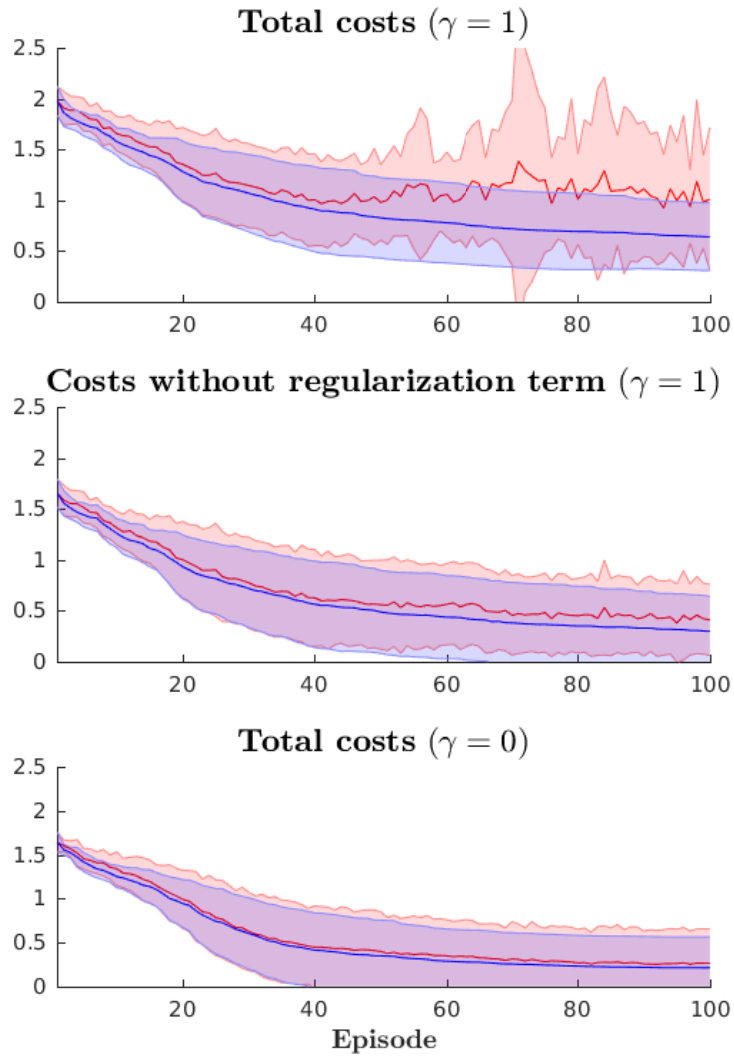


Figure 3.5: Results for compensating external forces with COMAN: top with regularization, middle with regularization but regularization term subtracted from total costs, bottom without regularization. The costs of the best roll-out per episode are shown (red) and the overall best roll-out (blue). Mean and standard deviation for all twenty trials are plotted.

3.3 Learning Soft Priorities for Compensating External Forces

The soft prioritization approach from [26] was initially proposed for the COMAN robot and control objectives were tested accordingly, however, in isolation. Here the next step towards full application of the approach is provided by simulating the 11 DOF upper body using and simplifying the Robotran simulation model for COMAN's whole-body [61]. Assuming the residual body to be static, the COMAN arm and upper body are modeled respectively to be attached to a fixed base. All described experiments are performed through the dynamics simulation available in Robotran [60], considering precise actuator dynamics for the 11 DOF upper body [61].

This section applies stochastic policy search as proposed in Sec. 3.1 to learn a policy which is capable of compensating external forces while maintaining a certain joint angle configuration. Note that COMAN's compliant elements provide the possibility for smooth and flexible movements.

COMAN is initialized with its home posture, standing straight in a minimum effort configuration except for the right arm, analogous to Fig. 3.4. The arm is bent such that the robot has a large manipulability but also has to compensate for gravity. The robot starts at rest, i.e.

initial joint velocities $\dot{\mathbf{q}} = \mathbf{0}$ and accelerations $\ddot{\mathbf{q}} = \mathbf{0}$. For all joints, a 2 rad/s velocity limit is defined.

The robot is simulated in each roll-out for a duration of 5 seconds while applying external forces to the CoM of the right forearm (most distal body part of the kinematic chain). For the sake of reproducibility, disturbances are represented as a combination of smooth random trajectories by employing four independent, one-dimensional DMPs [62] with random parameters of the function approximator. The DMP trajectories model the amplitude of the external force and the x-, y- and z-direction of the force vector over time. In each roll-out during policy search, new external forces with an average intensity of 10 N over time are sampled. Note that the robot does not know the disturbance profile. The external forces shall model a shaking disturbance caused by a human. In this section, no goal-variations such as different force primitives are evaluated to reduce simulation time.

All control objectives are applied except the end-effector controller because no reaching or tracking is required, resulting in a search space dimension of $K = 6$. According to [57], $4 + \lfloor 3\,log(K) \rfloor = 9$ roll-outs are evaluated per episode. Manually tuned initial values for $\boldsymbol{\lambda}_{init}$ are used to speed up convergence (Tab. 3.2). The initial standard deviation of CMA-ES is related to the initial policy parameters and treated as a vector with $0.2\,\boldsymbol{\lambda}_{init}$.

The cost function is chosen as a weighted combination of several criteria. Considering the overall high-level goal of compensating external forces, a natural criterion is the deviation of the end-effector $\|\mathbf{x}(t) - \mathbf{x}(t-1)\|$ between two discrete time steps $t-1$ and t and the deviation of the joint angles from their previous configuration $\Delta q(t) = \|\mathbf{q}(t) - \mathbf{q}(t-1)\|$. The latter is normalized by the number of joints D to allow comparison between different robot models.

In each time step t the cost $c(t)$ is computed as

$$c(t) = \alpha\ \|\mathbf{x}(t) - \mathbf{x}(t-1)\|\,(t) + \frac{\beta}{D}\,\Delta q(t) + \frac{\gamma}{D}\,\sum_{i=1}^{K} \lambda_i\ \|\boldsymbol{\tau}_i(t)\|\,, \tag{3.5}$$

with factors $\alpha = 5000$, $\beta = 8000$ and when incorporating regularization $\gamma = 1$, otherwise $\gamma = 0$. Costs are summed from multiple criteria neglecting different units. Note that α and β are chosen based on preliminary experiments.

If the generated movement exceeds joint angle or velocity limits, the simulation is stopped prematurely, and a fixed cost c_{max} is assigned to this roll-out. Otherwise, the resulting cost c is computed by integrating $c(t)$ over time.

The policy search is repeated twenty times with and without regularization term applying different random generator initializations to investigate the reliability of the overall learning framework. Tab. 3.2 lists the optimized priorities after 100 episodes. The optimization processes were able to detect the important control objectives joint momentum (JM) and joint velocity limit avoidance (JVLP), which both minimize joint velocities. The optimization for $\gamma = 0$ and $\gamma = 1$ end up with different priorities. As expected, when taking the regularization term into account ($\gamma = 1$), the gains for the irrelevant control objectives are considerably lower, and they contribute less to the overall motion.

Fig. 3.5 top shows the resulting costs per episode when incorporating regularization ($\gamma = 1$) into the cost function and Fig. 3.5 bottom when optimizing without regularization term ($\gamma = 0$). Fig. 3.5 top starts in the first episode with a higher initial cost because of the additional regularization criterion which is taken into account. In both cases the cost values decrease fast during the first episodes and then slowly converge, showing consistent behavior. This is as expected because the cost function (Eq. 3.5) is bounded from below (cost $c >= 0$) and it becomes increasingly difficult to reduce costs.

In Fig. 3.5 middle the combined cost terms for end-effector and joint angle deviations are plotted for the experiment with regularization. Comparing Fig. 3.5 middle and bottom reveals that slightly lower costs are obtained without regularization. These solutions compensate the external forces better, but also apply higher torques caused by considerable higher priorities as

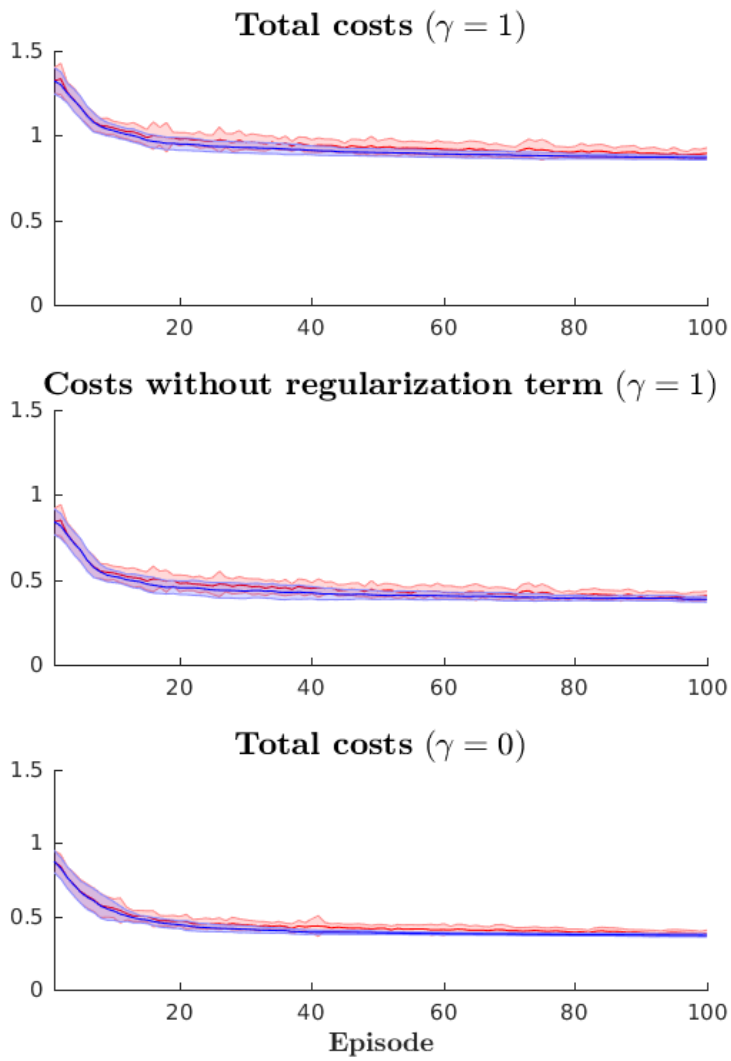


Figure 3.6: Results for the trajectory tracking experiment with COMAN: top with regularization, middle with regularization but regularization term subtracted from total costs, bottom without regularization. The costs of the best roll-out per episode are shown (red) and the overall best roll-out (blue). Mean and standard deviation for all twenty trials are plotted.

listed in Tab. 3.2. This is plausible because the energy consumption of the control objectives is not regarded in the cost function for $\gamma = 0$ and therefore does not affect the optimization process.

3.4 Learning Soft Priorities for Trajectory Tracking

In the next experiment, the optimization framework is applied to find optimum priorities for tracking a predefined 3D task-space trajectory by superimposing all control objectives mentioned above. Again the 11 DOF COMAN upper body is chosen, starting with the robot being in the same initial configuration as before. Gravity is the only external force acting upon the robot. The robot is simulated for a duration of 7 seconds in each roll-out. The end-effector objective controls the right forearm tip. A minimum jerk trajectory in 3D task-space is provided (see Fig. 3.4). This trajectory was fed preliminary to an inverse kinematics solver with 140 points equally distributed in time to guarantee that the trajectory can be tracked without violating joint angle and velocity constraints.

The cost is obtained again as a combination of several criteria. Considering a trajectory tracking goal, a natural criterion is the deviation of the end-effector from the desired target trajectory $\|\mathbf{x}_{\text{des}}(t) - \mathbf{x}_{\text{cur}}(t)\|$. Because the end-effector should follow a minimum jerk trajectory and in order to foster smoother motions, a minimum jerk criterion is accommodated. For this purpose, the absolute jerks $j(t) = \|\ddot{\mathbf{x}}(t) - \ddot{\mathbf{x}}(t-1)\|$ are incorporated. Additionally, the same regularization term is added as in the previous section.

In each time step t the cost

$$c(t) = \alpha \, \|\mathbf{x}_{\text{des}}(t) - \mathbf{x}_{\text{cur}}(t)\| \, (t) + \frac{\beta}{D} \, \|j(t)\| + \frac{\gamma}{D} \sum_{i=1}^{K} \lambda_i \, \|\boldsymbol{\tau}_i(t)\| \,, \tag{3.6}$$

is computed with factors $\alpha = 25$, $\beta = 0.06$ and when applying regularization $\gamma = 1$, otherwise $\gamma = 0$. The total cost of a roll-out is computed by integrating the cost for all discrete time steps during simulation, except when joint limits are violated as in the previous section. Again this cost function (Eq. 3.6) has the lower bound $c >= 0$.

The optimization process evaluates ten roll-outs per episode. It is started with the same initial priorities as in the experiment above and stopped after 100 episodes. The results of the optimization are shown for twenty repetitions in Fig. 3.6 top with regularization ($\gamma = 1$) and Fig. 3.6 bottom without ($\gamma = 0$). The costs decrease in the beginning and then converge. Fig. 3.6 middle shows the experiment with regularization ($\gamma = 1$) but the cost with subtracted regularization term, for better comparison to the experiment without regularization. Tab. 3.3

Table 3.2: Soft priorities and resulting total torque percentages per objective for compensating external forces: initial policy (left), optimized policy without regularization term (middle) and with regularization term (right). The priorities values represent the median over all twenty trials.

Objective	Initial		without R. ($\gamma = 0$)		with R. ($\gamma = 1$)	
	$\boldsymbol{\lambda}$	$\boldsymbol{\tau}$	$\boldsymbol{\lambda}$	$\boldsymbol{\tau}$	$\boldsymbol{\lambda}$	$\boldsymbol{\tau}$
ME	0.2	33.8 %	0.0	0.0 %	0.0	0.0 %
JM	0.5	41.9 %	368.2	72.0 %	281.2	82.4 %
$JALP$	10.0	1.6 %	3.7	3.3 %	0.0	0.0 %
$JALD$	0.5	0.5 %	2.0	3.2 %	0.7	1.4 %
$JVLP$	15.0	22.1 %	278.2	21.4 %	136.8	16.1 %
$JVLD$	0.1	0.1 %	1.2	0.1 %	0.8	0.1 %

Table 3.3: Soft priorities and resulting total torque percentages per objective for tracking a Cartesian trajectory: initial policy (left), optimized policy without regularization term (middle) and with regularization term (right). The priorities values represent the median over all twenty trials.

Objective	Initial		without R. ($\gamma = 0$)		with R. ($\gamma = 1$)	
	$\boldsymbol{\lambda}$	$\boldsymbol{\tau}$	$\boldsymbol{\lambda}$	$\boldsymbol{\tau}$	$\boldsymbol{\lambda}$	$\boldsymbol{\tau}$
ME	0.2	25.4 %	0.1	48.7 %	0.1	53.1 %
JM	0.5	16.6 %	0.0	0.0 %	0.7	7.0 %
$JALP$	10.0	16.7 %	5.6	9.2 %	0.0	0.0 %
$JALD$	0.5	2.4 %	0.1	1.1 %	0.0	0.8 %
$JVLP$	15.0	4.9 %	0.4	0.5 %	0.0	0.0 %
$JVLD$	0.1	0.1 %	0.4	0.1 %	0.4	0.1 %
EEP	15.0	30.8 %	375.6	40.3 %	365.8	39.0 %
EED	0.1	3.1 %	6.8	0.1 %	5.4	0.0 %

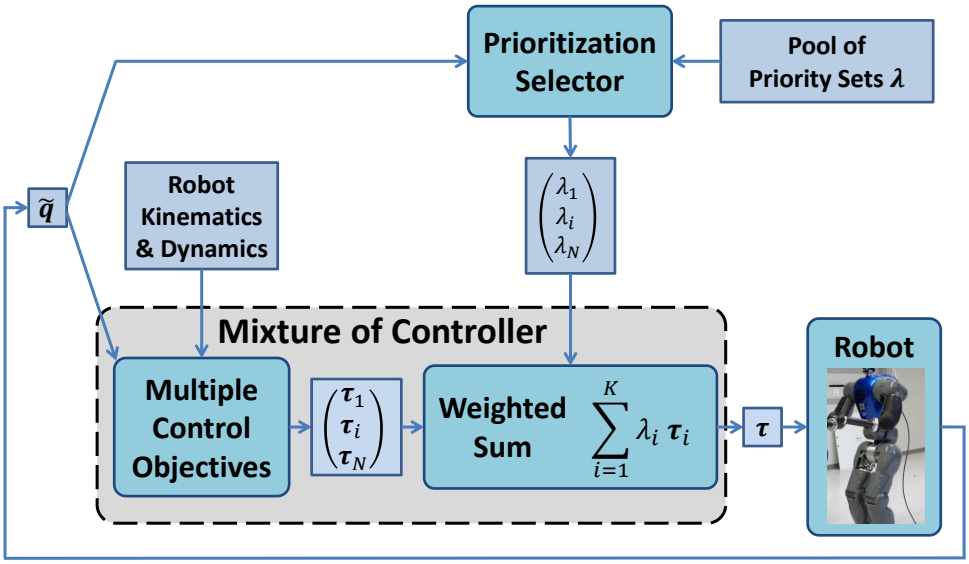


Figure 3.7: Soft prioritization between tasks extended from [26]. Lower-part: A multitude of controllers pursue different objectives and compute torques $\boldsymbol{\tau}_i$ which are superimposed using a set of scalar weights $\boldsymbol{\lambda}$ representing priorities. Upper-part: The policy selector is responsible for task-priority rearrangement. It chooses a prioritization (a policy) depending on the sensory information provided by the robot and can blend between solutions previously learned.

lists the optimized priorities. Note that the minimum effort and end-effector controllers are predominant to fulfill the desired overall high-level goal.

3.5 Continuous Priority Rearrangement with Soft Priorities

The soft hierarchy approach was originally designed to flexibly impose and change priorities during motion execution, however, this was not shown in [26]. This feature is difficult to achieve within the Stack-of-Tasks approach which relies on a strict predefined hierarchy and idempotent projectors.

Online continuous task-priority rearrangement is implemented by providing a simple strategy to blend priorities. Fig. 3.7 illustrates the system overview. A policy selector component is employed which sets the priorities in order to deal with current environmental conditions based upon a pool of previously optimized policies. For dynamically switching between two different policies $\boldsymbol{\lambda}_A$ and $\boldsymbol{\lambda}_B$, the corresponding priorities are adapted according to

$$\boldsymbol{\lambda}(t) = (1 - \sigma(t))\ \boldsymbol{\lambda}_A + \sigma(t)\, \boldsymbol{\lambda}_B, \tag{3.7}$$

where $\sigma(t)$ is monotone increasing with $0 <= \sigma(t) <= 1$. A naive solution implementing $\sigma(t)$ is to choose a hard jump at a specific time step t_*:

$$\sigma(t) = \begin{cases} 0, & \text{if } t < t_* \\ 1, & \text{otherwise.} \end{cases} \tag{3.8}$$

A sigmoid function is proposed to smoothly blend priorities within a short phase

$$\sigma(t) = \frac{1}{1 + e^{-k\,(t - t_*)}},\ k > 0\,. \tag{3.9}$$

The experiment is designed as follows: The robot starts following the predefined trajectory (red line in Fig. 3.4) as in Sec. 3.4 using the optimized priorities for trajectory tracking. After

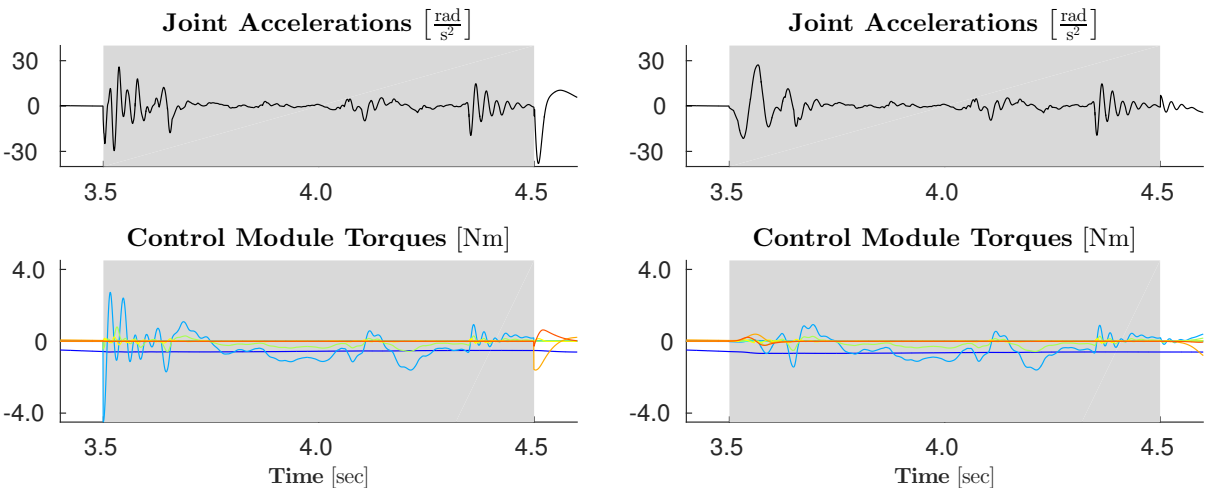


Figure 3.8: Generated motion for task-priority rearrangement by blending between two sets of priorities: left when applying a hard switch-over (Eq. 3.8) and right when applying the sigmoid function (Eq. 3.9). The accelerations and all K controller torques are shown for the right shoulder roll joint. Disturbances are applied in the interval from 3.5 to 4.5 seconds (shaded area). For visualization purposes only the relevant time window including the task priority-switches is shown.

3.5 seconds external forces are applied to the right forearm similar to Sec. 3.3. This example assumes a perfect classification of the actual level of disturbance. The policy selector recognizes this change in the environment and causes a task-priority rearrangement according to Eq. 3.7. After perturbing the robot for one second, the external forces are turned off, which again is recognized by the policy selector component. The priorities are blended dynamically back to continue tracking the desired trajectory, Here both implementations of $\sigma(t)$ are evaluated and compared with each other.

The joint accelerations and all K controller torques of the right shoulder roll joint are plotted exemplary in Fig. 3.8. When applying external forces, the tracking is stopped to better compensate the disturbances. The adaptation of task-priorities $\boldsymbol{\lambda}_A$ to $\boldsymbol{\lambda}_B$, caused by the hard switch (Eq. 3.8) is abrupt and results in higher torques and jerky movements, while the adaptation generated by the sigmoid function (Eq. 3.9, $k = 0.06$) is very smooth. The end-effector movement when applying the sigmoid function is visualized as a blue line in Fig. 3.4. As can be seen, the red target trajectory is tracked precisely. The duration of the rearrangement can be adjusted according to user requirements.

3.6 Comparison with state-of-the-art

The proposed approach in this chapter generalizes earlier approaches to optimize a single goal, e.g. [1]. A similar approach to robust generalization was proposed in [63], without consideration of dynamic properties of the robot. All priorities are equal, and stochastic optimization is employed to iteratively adapt task trajectories such that tasks can be fulfilled sequentially. Related to the proposed approach is also the work on learning coefficients for a mixture of several motion primitives presented in [64]. The authors in [12] sum multiple reference torques from various control objectives. However, the controller weights are not optimized. It is argued that controllers are carefully designed to avoid undesired and undefined competitions of control objectives. In contrast, with the proposed approach in this chapter, there is no need to consider such restrictions. Task priority rearrangement with soft priorities has been demonstrated in [25], however, relying on manually tuned priorities.

Recently, more approaches have been proposed utilizing smooth priorities between multiple task objectives building on top of my work published in [55] and [56]: Time-dependent soft priorities are learned in [65], parameterizing the weights in the torque superposition by radial basis functions. Optimal evolution of task priorities over time is then automatically tuned employing CMA-ES. The approach is further extended in [66] to ensure that constraints are never violated. [65] and [66] require an external cost function which is not the case for the approaches described in [67–69]. Given task trajectories with some variance, [67] proposes time-dependent weights which are regulated based on that variance. The task priority is at a maximum when the variance is small, and it diminishes where the variance is high. In this approach variance is encoded employing Gaussian kernels, where kernel centers are predefined along the trajectory. [68, 69] extend that idea exploiting variance extracted from multiple human demonstrations. Time-dependent priorities are learned probabilistically without requiring the previous definition of a cost function. In [68] velocity-controllers are exploited while [69] considers torque-controllers.

Another open question was, whether the soft prioritization becomes infeasible with an increasing number of control objectives applied to robots with many joints. [9] showed that it is possible to simultaneously control multiple humanoids coupled through interaction as a single multi-body system. A high number of objectives are tracked without employing nullspace projections. This approach formulates the soft prioritization as a quadratic program which allows to incorporate inequality constraints more accurately. For this approach, stability in terms of solution existence, uniqueness, continuity, and robustness to perturbations is proven in [7].

The interested reader is also referred to [70], where different approaches on learning soft task priorities have been reviewed recently.

3.7 Conclusions

In the Stack-of-Tasks a strict prioritized task hierarchy has to be designed by the developer based on prior knowledge, which is often not easy. The main advantage of the soft prioritization approach for multi-objective motion generation is the ability to impose smooth priorities for the underlying torque-controllers by defining a set of scalar weights representing priorities. This chapter establishes a general and repeatable method for automated learning suitable priority sets. The proposed optimization framework combines the soft prioritization scheme and CMA-ES. A user-specified cost function guides the optimization process to learn a policy for a dedicated high-level goal. This chapter demonstrates learning suitable priorities when reaching for targets, compensating external forces and tracking a 3D task-space trajectory. The proposed approach was demonstrated with the compliant humanoid robot COMAN in simulation.

The framework accounts for variability by evaluating multiple high-level goal variations such as different reaching targets in each roll-out of priorities. The ability to transfer the learned policy to new goals, which were not part of the optimization process, was demonstrated. This chapter shows that the optimized policy can generalize, for example to previously unknown reaching targets.

Furthermore, it is proposed to regularize the weights in the torque mixture. Because only a subset of all controllers may be relevant depending on the overall high-level goal, regularization detects the control objectives which are not relevant and reduces the corresponding amplitudes. In comparison to standard approaches in pattern classification and regression approaches, regularization cannot be implemented directly based on the level of weights for the superposition and a more sophisticated solution is required. The advantage of incorporating a regularization term in the cost function was shown.

Instead of tuning the whole set of priorities manually, fewer and better interpretable meta-parameters for automatic tuning of priorities have to be chosen. The search for good priorities from scratch cannot be done directly on a real-world robot system due to the risk of damaging

the robot. A malfunctioning set of priorities may result in disobeying hardware constraints or self-collisions. Additionally, many different roll-outs need to be executed which would be too time-consuming on a real robot. This approach avoids such problems by testing new sets of priorities in a simulation environment with an accurate dynamics simulation.

Finally, an example how to react to environmental changes by adapting priorities online was provided. When applying external forces suddenly, e.g. by a human interacting with the robot, the system reacts immediately and causes a smooth and continuous task-priority rearrangement for the underlying tasks. Policy selection is based upon a pool of previously learned priorities.

To the authors knowledge, all attempts to learn priorities reported in literature rely on soft prioritization with weights due to the continuous parametrization. However, for some applications a combination of both strict and soft priorities may be required. For this reason, the next chapter is devoted to prioritization schemes that combine advantages from both strict and soft prioritization into a single, continuous parametrization that facilitates learning advanced priorities.

Chapter 4

Managing Interference between Tasks

Both strict and soft prioritization approaches have their own merits. The main advantage of soft prioritization is that smooth adaptation of task priorities online is straightforward at the cost of uncontrolled interference – also called coupling – between tasks. In contrast, the alternative dynamically consistent Stack-of-Tasks approach fully decouples tasks, but continuous priority rearrangement is not trivial. Combining both prioritization schemes is a hot topic in robotics nowadays.

This chapter makes several major contributions to formally analyze interference between tasks with the goal to combine the benefits from both soft and strict prioritization into a novel more general approach. To this aim, this chapter first formalizes projection shaping and operational space task shaping generalizing previous work. Next, meaningful shaping operators are devised. Furthermore, the damped least squares technique is reformulated to analyze the effect of the regularization term. An illustrative 2 DOF example demonstrating shaping is discussed in detail. Then, shaping is applied within SoT to enable deactivation of single task-space dimensions as well as task insertion and removal. Finally, a novel approach referred to as dynamically consistent Generalized Hierarchical Control (dynGHC) is proposed which allows to define strict or soft priorities between each pair of tasks separately. It further enables to continuously rearrange priorities online and thereby allows to switch between the strict and soft scheme. This chapter highlights that tasks may interfere for two different reasons: soft priorities generate desired coupling between a set of tasks and inertia coupling introduces undesired accelerations in the operational spaces of other tasks. The proposed DynGHC isolates both effects and eliminates the latter in comparison to the existing statically consistent GHC approach. Main parts of this chapter have been published previously in [71] and the proposed DynGHC approach is currently under review for the *International Conference on Robotics and Automation, 2019*.

4.1 Shaping as a Means to study Interference between Tasks

4.1.1 Motivation

This chapter begins by introducing soft prioritization with matrix-priorities $\mathbf{T}_i$ in task-space, extending soft prioritization with scalar weights given in Eq. 2.30:

$$\boldsymbol{\tau} = \sum_{i=1}^{K} \mathbf{J}_i^T \mathbf{T}_i \mathbf{F}_i \,. \tag{4.1}$$

This formulation enables versatile modification of operational space tasks (e.g. deactivation of task dimensions) and degenerates to Eq. 2.30 for $\mathbf{T}_i = \lambda_i \, \mathbf{I}$.

Note that previous works proposed modifications of the Jacobian as an approximation: in near-singularity configurations, the critical singular values are zeroed [72]; in contact situations, rows of the end-effector Jacobian are zeroed to represent the constraint [54]. This thesis refers to those approaches as *shaping operational space tasks*. This chapter shows that such specific modifications can be obtained with the proposed more general formulation as special cases.

In robotics, related works propose modified projection matrices which are not idempotent, in contrast to projectors described in Sec. 2.2, and often ignore the errors introduced as pointed out in [18]. In contrast to operational space task shaping, this thesis refers to such matrix operators as *shaped projectors* because they approximate proper idempotent projections: for example scalar scaling [73], successive nullspace projection [14], DLS regularization [20], or shaping a specific orthonormal basis in [21] and [43].

Inconsistencies may occur when shaping operational space tasks and/or nullspace projections: Uncontrolled DOFs may remain in such control schemes. Moreover, tasks which were supposed to be decoupled may interfere with each other due to the approximative shaping character.

4.1.2 Projection Shaping

The Jacobian as a function of joint angles is defined as a continuous mapping but rank-changes result in dimension-shifts between its range and nullspace [22], [23]. When projecting a torque control objective onto this range or nullspace, such discontinuities are reflected in the commanded torques which is disadvantageous. Thus, a technique to continuously shift DOFs between range and nullspace is desired. In the following, the necessary analytical framework is derived to achieve this.

Given an orthonormal basis $\mathbf{B}$ (with $\mathbf{B}^T\mathbf{B} = \mathbf{I}$), the orthogonal projections onto the range and nullspace of $\mathbf{B}$ are defined by $\mathbf{H} = \mathbf{B}\,\mathbf{B}^T$ and $\mathbf{N} = \mathbf{I} - \mathbf{B}\,\mathbf{B}^T$ respectively [38, Eq. 5.13.4]. With a little abuse of nomenclature, define a *shaped projector* which is not necessarily idempotent and approximates an orthogonal projection as

$$\mathbf{H}^{'} = \mathbf{B}\mathbf{T}\mathbf{B}^T \text{ and its complement } \mathbf{N}^{'} = \mathbf{I} - \mathbf{B}\mathbf{T}\mathbf{B}^T, \tag{4.2}$$

where $\mathbf{B}$ is an orthonormal basis, and $\mathbf{T}$ shapes the original projection. Note that Eq. 4.2 is continuous with respect to $\mathbf{T}$. Mathematically, $\mathbf{H}^{'}$, $\mathbf{N}^{'}$ constitute endomorphisms, viz., linear maps from a vector space to itself. Referring to $\mathbf{H}^{'}$, $\mathbf{N}^{'}$ as shaped projectors follows a widely used practice in robot control where the term "projection" is loosely used for non-idempotent matrix operators as well which merely approximate projections. Note that $\mathbf{H}^{'}$, $\mathbf{N}^{'}$ introduce coupling between shaped range and nullspace which is typically neglected in a control scheme. Depending on the choice of $\mathbf{T}$, not all degrees of freedom are affected. Some DOFs are "shared" between shaped range and nullspace while the rest are clearly assigned to either of the spaces. However, any torque vector can uniquely be decomposed into two parts belonging to the shaped range and nullspace: $\boldsymbol{\tau} = \boldsymbol{\tau}_{\mathbf{H}'} + \boldsymbol{\tau}_{\mathbf{N}'}$. Projecting a primary task onto such shaped range and a secondary task onto the complementary shaped nullspace will result in interference between both tasks exactly in these shared dimensions while both tasks are decoupled in the other dimensions.

In general, neither $\mathbf{H}^{'}$, nor $\mathbf{N}^{'}$ are proper projection matrices. However, if $\mathbf{T}$ is idempotent, so are $\mathbf{H}^{'}$ and $\mathbf{N}^{'}$: $(\mathbf{H}^{'})^2 = \mathbf{B}\mathbf{T}\mathbf{B}^T\mathbf{B}\mathbf{T}\mathbf{B}^T = \mathbf{B}\mathbf{T}\mathbf{B}^T = \mathbf{H}^{'}$. Then, they constitute proper projections and $\mathbf{H}^{'}$, $\mathbf{N}^{'}$ project onto the range and nullspace of $\mathbf{BT}$ respectively.

This thesis restricts $\mathbf{T}$ to be a diagonalizable matrix with eigendecomposition $\mathbf{T} = \mathbf{O}\mathbf{A}\mathbf{O}^{-1}$ where $\mathbf{O}$ is an invertible matrix and $\mathbf{A}$ a diagonal matrix described below. This definition clearly generalizes the approaches described in [21] and [43]. Further evaluating even more general $\mathbf{T}$ is subject to future work.

Let the diagonal elements of $\mathbf{A}$ be $d_{i,i} \in [0, 1]$. Owing to the structure of $\mathbf{T}$, and hence also $\mathbf{H}^{'}$, its eigenvalues are identical with the diagonal elements of $\mathbf{A}$. Therefore, both $\mathbf{T}$ and

the shaped projector $\mathbf{H}^{'}$ scale vectors by $d_{i,i}$ along the i-th eigenvectors in $\mathbf{O}$ or $\mathbf{BO}$ respectively. Hence, $d_{i,i} = 0$ fully removes the i-th dimension and $d_{i,i} = 1$ represents a proper projection along the i-th eigenvector. Therefore, interference between shaped range and nullspace projection introduced by $\mathbf{T}$ in Eq. 4.2 only affects the directions corresponding to $d_{i,i} \in]0, 1[$ in $\mathbf{A}$. As a result, interpolating $\mathbf{T}$ between idempotent matrices $\mathbf{T}_1$, $\mathbf{T}_2$, allows to deal with rank-changes between the range of $\mathbf{BT}_1$ and $\mathbf{BT}_2$ continuously. Sec. 4.2 presents an illustrative 2 DOF example.

4.1.3 Operational Space Task Shaping

For some applications it may be helpful to modify the task at runtime, i.e. deactivating desired task-space dimensions. Considering a task-space transformation matrix $\mathbf{T}$ a *shaped operational space task* is defined as

$$\boldsymbol{\tau} = \mathbf{H}\,\mathbf{J}^T\mathbf{T}\,\mathbf{F} = \left(\mathbf{J}^T\left(\mathbf{J}^+\right)^T\right)\mathbf{J}^T\mathbf{T}\,\mathbf{F}. \tag{4.3}$$

Interpolating $\mathbf{T}$ between $\mathbf{T}_1$, $\mathbf{T}_2$ will result in continuous shaping of the operational space task limiting the torque derivative, which can be interpreted from three perspectives:

i) modification of the wrench $\mathbf{F}^{'} = \mathbf{T}\,\mathbf{F}$ with $\boldsymbol{\tau} = \mathbf{H}\,\mathbf{J}^T\mathbf{F}^{'}$,

ii) modification of the Jacobian $\mathbf{J}^{T\,'} = \mathbf{J}^T\mathbf{T}$ with $\boldsymbol{\tau} = \mathbf{H}\,\mathbf{J}^{T\,'}\mathbf{F}$,

iii) modification of the projector onto the range $\mathbf{H}^{'} = \mathbf{J}^T\mathbf{T}(\mathbf{J}^+)^T$ with $\boldsymbol{\tau} = \mathbf{H}^{'}\,\mathbf{J}^T\mathbf{F}$.

The first interpretation is meaningful when considering control of a single task wrench which is modified to fulfill specific criteria. The second interpretation is helpful when studying the effect of $\mathbf{T}$ on the singular values or the rank of $\mathbf{J}$. Third, $\mathbf{H}^{'}$ represents a shaped projector according to Eq. 4.2. It constitutes an endomorphism, viz., a linear map from the vector space of torques to itself which can be interpreted in three steps: Mapping an admissible torque vector $\boldsymbol{\tau} = \boldsymbol{\tau}_\mathbf{H} + \boldsymbol{\tau}_\mathbf{N}$ onto a task-space wrench by $(\mathbf{J}^+)^T$ thereby "losing" the part which belongs to the nullspace ($\boldsymbol{\tau}_\mathbf{N}$), projecting the wrench onto a subspace of the range of $\mathbf{J}^T$ by multiplication with $\mathbf{T}$ and finally mapping the new wrench onto a joint-space torque vector $\boldsymbol{\tau}^{'}$.

Hence, through these steps a shaped projection Eq. 4.2 can also be interpreted as operational space task shaping Eq. 4.3. In fact, the error in task-space introduced by a shaped projection can geometrically be interpreted when being able to formulate it as $\mathbf{H}^{'}$. The error is then equivalent to the modification of the wrench $\mathbf{F}$ to $\mathbf{F}^{'}$ by multiplication with $\mathbf{T}$. It is thus obvious that the coupling introduced by $\mathbf{T}$ does not necessarily affect all task-space dimensions when applying operational space task shaping. Hence, choosing $\mathbf{T}$ carefully, the obtained solution is still optimal in some dimensions.

4.1.4 Shaping applied within Multi-Objective Prioritization

Employing both shaped operational space tasks with $\mathbf{T}_i$ (Eq. 4.3) and shaped nullspace projections $\mathbf{N}_i^{'}$ (Eq. 4.2) simultaneously, combines and extends strict and soft prioritization:

$$\boldsymbol{\tau} = \sum_{i=1}^{K} \mathbf{N}_i^{'}\,\mathbf{J}_i^T\mathbf{T}_i\mathbf{F}_i\,. \tag{4.4}$$

Continuously shaping $\mathbf{T}_i$ and $\mathbf{N}_i^{'}$ allows to rearrange task priorities without discontinuities in the commanded torques. The main question is how to choose $\mathbf{T}_i$ reasonably. The following subsections devise meaningful matrix priorities and shape corresponding nullspace projections such that interference between tasks is managed within a subset of all DOFs. Note that it is already difficult to find a suitable set of scalar priorities for soft prioritization applying learning

techniques as demonstrated in the previous chapter because an appropriate cost function has to be defined and the optimization has to deal with local minima. Hence, employing high-dimensional policy search processes to learn matrix-priorities will be even more difficult and time-consuming. This section lays the foundations for simplifying the policy search by devising suitable transformation matrices $\mathbf{T}_i$ with a low number of parameters. To this end, the effect of $\mathbf{T}_i$ is analytically studied with respect to different orthonormal bases obtained by decomposing the Jacobian.

4.1.5 Special Orthonormal Bases

In the following, consider a Jacobian $\mathbf{J} \in \mathbb{R}^{H\times D}$ with rank $r \leq H \leq D$, i.e. the Jacobian is not necessarily full-rank. Employing the singular value decomposition (SVD) [20], the Jacobian transpose is factorized as

$$\mathbf{J}_{\text{SVD}}^T = \mathbf{V} \begin{bmatrix} \mathbf{S} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{U}^T = \mathbf{V}_{1:r} \begin{bmatrix} \mathbf{S} & \mathbf{0} \end{bmatrix} \mathbf{U}^T, \tag{4.5}$$

where $\mathbf{S} \in \mathbb{R}^{r\times r}$ is square and holds the positive singular values sorted in descending order as diagonal elements and $\mathbf{U} \in \mathbb{R}^{H\times H}$, $\mathbf{V} \in \mathbb{R}^{D\times D}$ are orthonormal matrices. The columns of $\mathbf{U}$ of $\mathbf{V}$ are the left- and right-singular vectors of $\mathbf{J}$ and are respectively the eigenvectors of $\mathbf{J}\mathbf{J}^T$ and $\mathbf{J}^T\mathbf{J}$. The index $()_{1:r}$ denotes the selected columns for a truncated decomposition. The Moore-Penrose inverse of $\mathbf{J}_{\text{SVD}}^T$ is

$$\mathbf{J}_{\text{SVD}}^+ = \mathbf{V}_{1:r} \begin{bmatrix} \mathbf{S} & \mathbf{0} \end{bmatrix} \mathbf{U}^T \Big(\mathbf{U} \begin{bmatrix} \mathbf{S} & \mathbf{0} \end{bmatrix}^T \mathbf{V}_{1:r}^T \mathbf{V}_{1:r} \begin{bmatrix} \mathbf{S} & \mathbf{0} \end{bmatrix} \mathbf{U}^T \Big)^{-1}, \tag{4.6}$$

and can be further simplified because $\mathbf{V}_{1:r}^T\mathbf{V}_{1:r} = \mathbf{I}$, $\mathbf{U}^T\mathbf{U} = \mathbf{I}$

$$\mathbf{J}_{\text{SVD}}^+ = \mathbf{V}_{1:r} \begin{bmatrix} \mathbf{S}^{-1} & \mathbf{0} \end{bmatrix} \mathbf{U}^T. \tag{4.7}$$

Since $\mathbf{S}$ is diagonal, $\mathbf{S}^{-1}$ is also diagonal and easy to compute with reciprocals of entries of $\mathbf{S}$. The orthogonal range and nullspace projections $\mathbf{H}_{\text{SVD}}$, $\mathbf{N}_{\text{SVD}} = \mathbf{I} - \mathbf{H}_{\text{SVD}}$ are formulated without inverse operation

$$\mathbf{H}_{\text{SVD}} = \mathbf{V}_{1:r} \begin{bmatrix} \mathbf{S} & \mathbf{0} \end{bmatrix} \mathbf{U}^T\mathbf{U} \begin{bmatrix} \mathbf{S}^{-1} & \mathbf{0} \end{bmatrix}^T \mathbf{V}_{1:r}^T = \mathbf{V}_{1:r}\mathbf{V}_{1:r}^T. \tag{4.8}$$

A QR decomposition (QRD) [74] with pivoting of the Jacobian transpose results in

$$\mathbf{J}_{\text{QRD}}^T = \mathbf{Q} \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{P}^T\mathbf{I} = \mathbf{Q}_{1:r} \begin{bmatrix} \mathbf{R} & \mathbf{0} \end{bmatrix} \mathbf{P}^T\mathbf{I}\,, \tag{4.9}$$

where $\mathbf{Q} \in \mathbb{R}^{D\times D}$ is an orthonormal matrix and $\mathbf{R} \in \mathbb{R}^{r\times r}$ an upper triangular matrix. The permutation matrix $\mathbf{P} \in \mathbb{R}^{H\times H}$ ensures that the positive diagonal entries in $\mathbf{R}$ are in non-increasing order. The orthogonal identity matrix $\mathbf{I} \in \mathbb{R}^{H\times H}$ is added to highlight the similarity with SVD. The Moore-Penrose inverse of $\mathbf{J}_{\text{QRD}}^T$ is

$$\mathbf{J}_{\text{QRD}}^+ = \mathbf{Q}_{1:r} \begin{bmatrix} \mathbf{R} & \mathbf{0} \end{bmatrix} \mathbf{P}^T \Big(\mathbf{P} \begin{bmatrix} \mathbf{R} & \mathbf{0} \end{bmatrix}^T \mathbf{Q}_{1:r}^T \mathbf{Q}_{1:r} \begin{bmatrix} \mathbf{R} & \mathbf{0} \end{bmatrix} \mathbf{P}^T \Big)^{-1} \tag{4.10}$$

and simplifies, because $\mathbf{Q}_{1:r}^T\mathbf{Q}_{1:r} = \mathbf{I}$ and $\mathbf{P}^T\mathbf{P} = \mathbf{I}$

$$\mathbf{J}_{\text{QRD}}^+ = \mathbf{Q}_{1:r} \begin{bmatrix} \left(\mathbf{R}^T\right)^{-1} & \mathbf{0} \end{bmatrix} \mathbf{P}^T\mathbf{I}\,. \tag{4.11}$$

Likewise, with QRD, the orthogonal range and nullspace projections $\mathbf{H}_{\text{QRD}}$, $\mathbf{N}_{\text{QRD}} = \mathbf{I} - \mathbf{H}_{\text{QRD}}$ do not require an inverse operation

$$\mathbf{H}_{\text{QRD}} = \mathbf{Q}_{1:r} \begin{bmatrix} \mathbf{R} & \mathbf{0} \end{bmatrix} \mathbf{P}^T\mathbf{P} \begin{bmatrix} (\mathbf{R}^{-1})^T & \mathbf{0} \end{bmatrix}^T \mathbf{Q}_{1:r}^T = \mathbf{Q}_{1:r}\mathbf{Q}_{1:r}^T\,. \tag{4.12}$$

SVD and QRD provide two different bases to represent the end-effector task-space. While the basis $\mathbf{I}$ implicitly associated with QRD is independent of the joint configuration, with SVD, small changes in joint configuration might result in large changes of the basis associated with $\mathbf{U}$, especially in the neighborhood of kinematic singularities.

In the remainder of this thesis, the indices for $\mathbf{V}_{1:r}$ and $\mathbf{Q}_{1:r}$ are omitted for ease of reading. Furthermore, $\begin{bmatrix}\mathbf{S} & \mathbf{0}\end{bmatrix}$ and $\begin{bmatrix}\mathbf{R} & \mathbf{0}\end{bmatrix} \in \mathbb{R}^{r\times H}$ are considered as non-square $\mathbf{S}$, $\mathbf{R}$ with appropriate dimensions respectively, hence $\mathbf{S}^T \neq \mathbf{S}$.

4.1.6 Special Shaping Operators

The next contribution of this chapter is achieved by selecting $\mathbf{T}$ in several different ways. Through the proposed formulation there is a clear interpretation how the interference between shaped range and nullspace arises and how and in which dimensions projected tasks in the extended SoT Eq. 4.4 are still decoupled.

Consider first the simplest case $\mathbf{T} = \alpha\,\mathbf{I}$ with scalar α proposed by [73] as is the case for scalar priorities in Eq. 2.30, resulting in deactivation of all task-dimensions simultaneously $\mathbf{H}' = \alpha\,\mathbf{J}^T(\mathbf{J}^+)^T = \alpha\,\mathbf{V}\mathbf{V}^T = \alpha\,\mathbf{Q}\mathbf{Q}^T$ or $\mathbf{F}' = \alpha\,\mathbf{F}$. Hence, the choice of an orthonormal basis is irrelevant and the computationally most efficient solution can be chosen.

Next other specific choices for $\mathbf{T}$ are studied based on SVD and QRD. Consider $\mathbf{T} = \mathbf{U}\mathbf{A}\mathbf{U}^T$ with diagonal $\mathbf{A} \in \mathbb{R}^{H\times H}$ employing SVD ($\mathbf{J}^T_{\text{SVD}} = \mathbf{V}\mathbf{S}\mathbf{U}^T$). This choice results into:

i) $\mathbf{F}'_{\text{SVD}} = \mathbf{U}\mathbf{A}\mathbf{U}^T\mathbf{F}$. Evaluating the resulting wrench, the multiplication by $\mathbf{U}^T$ represents a transformation into another coordinate system where the wrench is modified by multiplying $\mathbf{A}$, before transforming it back into its original coordinate system by multiplication with $\mathbf{U}$.

ii) $\mathbf{J}^{T\,\prime}_{\text{SVD}} = \mathbf{V}\mathbf{S}\mathbf{A}\mathbf{U}^T$. Choosing diagonal entries $0 \leq a_{ii} \leq 1$ allows to continuously deactivate individual singular values of the Jacobian. This allows to deal with singularities [72].

iii) $\mathbf{H}'_{\text{SVD}} = \mathbf{V}\mathbf{S}\mathbf{A}\big(\mathbf{S}^T\mathbf{S}\big)^{-1}\mathbf{S}^T\mathbf{V}^T$. Choosing diagonal entries between One and Zero, the shaped range and nullspace interfere exactly in singular vectors i with $0 < a_{ii} < 1$ and are separated in singular vectors j with entries $a_{jj} \in \{0, 1\}$.

Note that $\mathbf{T} = \mathbf{U}\mathbf{A}\mathbf{U}^T$ allows to precisely regulate the activation of each singular vector direction but not specific task directions (associated with rows of the Jacobian $\mathbf{J}$. The latter is rather possible by choosing the QRD ($\mathbf{J}^T_{\text{QRD}} = \mathbf{Q}\mathbf{R}\mathbf{P}^T\mathbf{I}$) in the following. Considering a diagonal $\mathbf{T}{=}\mathbf{I}\,\mathbf{A}\,\mathbf{I}$ results in:

i) $\mathbf{F}'_{\text{QRD}} = \mathbf{A}\mathbf{F}$. This formulation represents a scaling operation of the wrench in its original coordinate system.

ii) $\mathbf{J}^{T\,\prime}_{\text{QRD}} = \mathbf{Q}\mathbf{R}\mathbf{P}^T\mathbf{A}\,\mathbf{I}$. Choosing diagonal entries between One and Zero allows to continuously deactivate the orthogonal task-directions of the Jacobian. The proposed approach here subsumes e.g. [54] as a special case.

iii) $\mathbf{H}'_{\text{QRD}} = \mathbf{Q}\mathbf{R}\mathbf{P}^T\mathbf{A}\mathbf{P}\big(\mathbf{R}^T\mathbf{R}\big)^{-1}\mathbf{R}^T\mathbf{Q}^T$. Choosing diagonal entries $0 \leq a_{ii} \leq 1$, the shaped range and nullspace interfere exactly in task-space directions i with $0 < a_{ii} < 1$ and are separated in task-space directions j for $a_{jj} \in \{0, 1\}$.

Note that a diagonal matrix $\mathbf{A}$ deactivates orthogonal vectors relative to the Jacobian reference frame, i.e. typically the robot base frame. Deactivation of any other direction is possible when applying $\mathbf{A}$ after transforming the wrench into a respective coordinate system and before transforming it back. This is equivalent to changing the reference frame of the Jacobian [75].

It is important to note that general transformations do not commute. When combining multiple shaping operations $\mathbf{T}_i$ in different coordinate systems choosing the order has a clear

impact: $\mathbf{T}_1\mathbf{T}_2 \neq \mathbf{T}_2\mathbf{T}_1$. For example when deactivating certain rows of the Jacobian and at the same time modifying singular values:

$$\mathbf{A}_{\text{QRD}} \left(\mathbf{U}\mathbf{A}_{\text{SVD}}\mathbf{U}^T\right) \neq \left(\mathbf{U}\mathbf{A}_{\text{SVD}}\mathbf{U}^T\right) \mathbf{A}_{\text{QRD}} \ . \tag{4.13}$$

Further note that a product of idempotent transformations $\mathbf{T}_i = \mathbf{T}_i^2$ is not idempotent in general: $\mathbf{T}^2 \neq \mathbf{T} = \sum \mathbf{T}_i$. In the remainder of this thesis such complex shaping operations are not considered and is part of future work.

4.1.7 Relation to Damped Least Squares

This subsection shows the equivalent relation between damped least squares (DLS) and shaped projectors as defined in Eq. 4.2. From this perspective, specific choices of $\mathbf{T}$ described above exploit a more sophisticated regularization term. DLS (Eq. 2.31) with regularization $\mathbf{Z} = z^2\,\mathbf{I}$ can be reformulated employing SVD when choosing

$$\mathbf{N}' = \mathbf{I} - \mathbf{V}\mathbf{A}\mathbf{V}^T \text{ with } \mathbf{A}^{-1} = \left(\mathbf{S}\,{}^1\!/_{z^2}\,\mathbf{S}^T\right)^{-1} + \mathbf{I} \ , \tag{4.14}$$

or with QRD

$$\mathbf{N}' = \mathbf{I} - \mathbf{Q}\mathbf{A}\mathbf{Q}^T \text{ with } \mathbf{A}^{-1} = \left(\mathbf{R}\,{}^1\!/_{z^2}\,\mathbf{R}^T\right)^{-1} + \mathbf{I} \ . \tag{4.15}$$

The derivation is provided in the Appendix A. These two formulations allow to study the coupling introduced w.r.t. shaping of the orthonormal bases $\mathbf{V}$ and $\mathbf{Q}$.

For $z = 0$ (no damping, equivalent to $\mathbf{A} = \mathbf{I}$) and $z = \infty$ (total damping, equivalent to $\mathbf{A} = \mathbf{0}$) full activation and full deactivation of all task-space directions is achieved respectively. Imposing DLS regularization effects all singular values and all task-directions. Instead, it is certainly better to continuously shape only critical singular values (for internal singularity configurations) or critical task-dimensions (for external task-space constraints).

4.1.8 Imposing a Weighting

The inverse of the joint-space inertia matrix $\mathbf{M} \in \mathbb{R}^{D\times D}$ is often employed as weighting matrix $\mathbf{W} = \mathbf{M}^{-1}$ to construct dynamically consistent multi-level hierarchies with the Stack-of-Tasks (Eq. 2.27), see for example [11], [14], [15]. The shaped operational space formulation with a weighted projection onto the range

$$\boldsymbol{\tau} = \mathbf{H}^{\mathbf{W}}\,\mathbf{J}^T\mathbf{T}\,\mathbf{F} \tag{4.16}$$

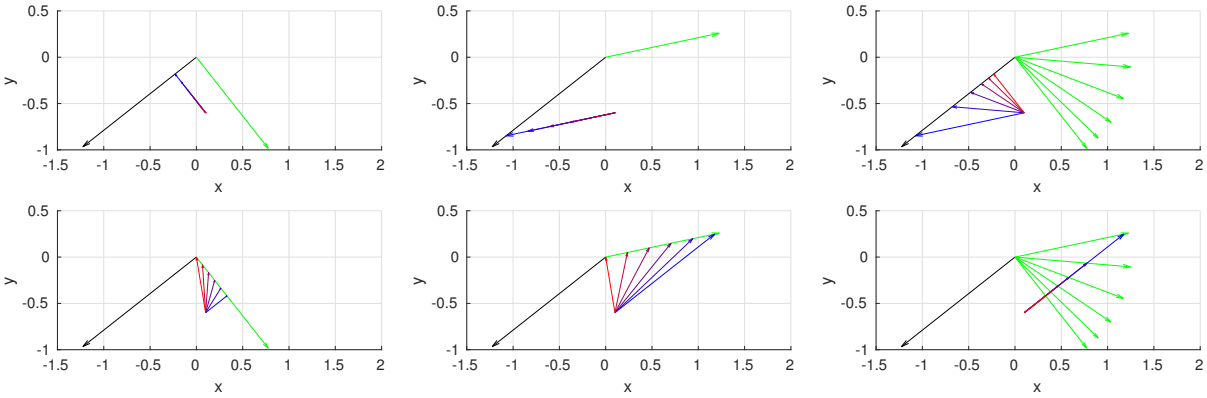
can be interpreted as $\boldsymbol{\tau} = \mathbf{H}^{\mathbf{W}\prime}\,\mathbf{J}^T\,\mathbf{F}$ with shaped projector $\mathbf{H}^{\mathbf{W}\prime} = \mathbf{J}^T\mathbf{T}\left(\mathbf{J}\mathbf{W}\mathbf{J}^T\right)^{-1}\mathbf{J}\mathbf{W}$, extending the definition Eq. 4.2 to the weighted case.

The wrench evaluates to $\mathbf{F} = \left(\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T\right)^{-1}\ddot{\mathbf{x}}$ in case operational space trajectories are represented as reference accelerations $\ddot{\mathbf{x}}$. Instead of shaping the wrench with Eq. 4.16, it is also possible to shape the reference acceleration

$$\boldsymbol{\tau} = \mathbf{H}^{\mathbf{M}^{-1}}\,\mathbf{J}^T\left(\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T\right)^{-1}\mathbf{T}\,\ddot{\mathbf{x}} \ , \tag{4.17}$$

which can be reformulated as $\boldsymbol{\tau} = \mathbf{H}^{\mathbf{M}^{-1}\prime}\,\mathbf{J}^T\,\mathbf{F}$ with $\mathbf{H}^{\mathbf{M}^{-1}\prime} = \mathbf{J}^T\left(\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T\right)^{-1}\mathbf{T}\mathbf{J}\mathbf{M}^{-1}$. Considering the effect $\boldsymbol{\tau}' = \mathbf{H}^{\mathbf{M}^{-1}\prime}\boldsymbol{\tau}$, a torque vector $\boldsymbol{\tau}$ is initially transformed into a task acceleration through the multiplication by $\mathbf{J}\mathbf{M}^{-1}$. Then, the acceleration is shaped by $\mathbf{T}$, successively converted into a wrench by $\left(\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T\right)^{-1}$ and finally mapped onto joint torques again by multiplication with $\mathbf{J}^T$. Analogous to the map $\mathbf{H}'$ in Sec. 4.1.3, the shaped projector $\mathbf{H}^{\mathbf{M}^{-1}\prime}$ constitutes an endomorphism which maps joint torques to joint torques intermediately converting them to task-space quantities.

The proposed formalism to continuously shift DOFs between range and nullspace can be extended to weighted projections with specific choices for deactivating singular vectors with SVD or shaping rows of the Jacobian with QRD. This is equivalent to DLS regularization within a weighted projection again. The proof is given in Appendix B.



a) shaping orthogonal projections, see Eq. 4.21

b) shaping oblique projections, see Eq. 4.22

c) different weighted projections, see Eq. 4.23

Figure 4.1: Projection shaping illustrated with a 2-DOF planar robot arm. The range space of $\mathbf{J}_x$ is visualized as black arrow in a-c. The orthogonal a) or oblique b and c) nullspace of $\mathbf{J}_x$ is visualized as green arrows. The top row shows shaped projections onto the range, the bottom row shows shaped nullspace projections for $\boldsymbol{\tau} = (0.1 \ \text{-}0.6)^T$. Intermediate arrows between red and blue visualize the smooth interpolation caused by shaped projection operations where red indicates $\alpha = 1$ and blue $\alpha = 0$. Note that $\boldsymbol{\tau} = \mathbf{H}_\alpha \boldsymbol{\tau} + \mathbf{N}_\alpha \boldsymbol{\tau}$. Imposing different weighting matrices $\mathbf{W}_\beta$ in c) the angle between range and complementary nullspace is varied: red indicates $\beta = 1$ and blue $\beta = \text{-}0.17$ to achieve a nullspace projection equivalent to a) and b).

4.2 Illustrative Example demonstrating Projection Shaping

This section visualizes how projection shaping allows to interpolate between two different idempotent matrices. Consider a planar robot arm with two rotational joints (first angle $\frac{\pi}{12}$ rad and second angle $\frac{\pi}{3}$ rad) and unit link length at rest, resulting in $\mathbf{J}_{\text{xy}} = \begin{pmatrix}\mathbf{J}_{\text{x}}^T & \mathbf{J}_{\text{y}}^T\end{pmatrix}^T$ with $\mathbf{J}_{\text{x}} = \begin{pmatrix}-1.22 & -0.97\end{pmatrix}$ and $\mathbf{J}_{\text{y}} = \begin{pmatrix}1.22 & 0.26\end{pmatrix}$. When controlling both x and y position of the end-effector, the projection onto the range employing QRD (Eq. 4.12) with $\mathbf{Q} = \begin{pmatrix}\mathbf{q}_1 & \mathbf{q}_2\end{pmatrix}$ becomes

$$\mathbf{H}_{\text{xy}} = \mathbf{Q}_{1:2}\mathbf{Q}_{1:2}^T = \mathbf{q}_1\mathbf{q}_1^T + \mathbf{q}_2\mathbf{q}_2^T = \mathbf{I} \ , \tag{4.18}$$

and there is no remaining non-trivial nullspace, i.e. $\mathbf{N}_{\text{xy}} = \mathbf{0}$. An exemplary torque vector $\boldsymbol{\tau} = (0.1 \ \text{-}0.6)^T$ is projected onto itself and onto $\mathbf{0}$ respectively (red arrows in Fig. 4.1a and b). Giving up control in y-direction when considering $\mathbf{J}_{\text{x}}$ only, results in orthogonal projectors, indicated by index $()_\perp$

$$\mathbf{H}_{\text{x},\perp} = \mathbf{Q}_1\mathbf{Q}_1^T = \mathbf{q}_1\mathbf{q}_1^T \text{ and } \mathbf{N}_{\text{x},\perp} = \mathbf{I} - \mathbf{H}_{\text{x},\perp} \ . \tag{4.19}$$

Hence, $\mathbf{H}_{\text{x},\perp}$ projects orthogonally onto the range of $\mathbf{J}_{\text{x}}^T$. And $\mathbf{N}_{\text{x},\perp}$ projects onto the orthogonal nullspace of $\mathbf{J}_{\text{x}}^T$, along the vector $\mathbf{J}_{\text{x}}^T$ (blue arrows in Fig. 4.1a). In this 2D example the range and nullspace of $\mathbf{J}_{\text{x}}^T$ correspond to lines and are visualized by black and green arrows respectively in Fig. 4.1a.

$\mathbf{N}_{\text{x},\perp}$ does not project onto the range of $\mathbf{J}_{\text{y}}^T$ but this is achieved employing an oblique projection, indicated by $()_\angle$

$$\mathbf{H}_{\text{x},\angle} = \mathbf{Q}_{1:2}\mathbf{R}\begin{pmatrix}1 & 0\\ 0 & 0\end{pmatrix}\mathbf{R}^{+}\mathbf{Q}_{1:2}^T \text{ and } \mathbf{N}_{\text{x},\angle} = \mathbf{I} - \mathbf{H}_{\text{x},\angle} \ . \tag{4.20}$$

This rangespace projector $\mathbf{H}_{\text{x},\angle}$ projects onto $\mathbf{J}_{\text{x}}^T$ (similarly to $\mathbf{H}_{\text{x},\perp}$) but along the vector $\mathbf{J}_{\text{y}}^T$. And the corresponding nullspace projector $\mathbf{N}_{\text{x},\angle}$ projects onto the space spanned by $\mathbf{J}_{\text{y}}^T$ along the vector $\mathbf{J}_{\text{x}}^T$ (blue arrows in Fig. 4.1b). The oblique range and nullspace of $\mathbf{J}_{\text{x}}^T$ correspond

to lines with non-orthognal angle and are visualized by black and green arrows respectively in Fig. 4.1b.

Abruptly switching from $\mathbf{H}_{\mathrm{xy}}$ to $\mathbf{H}_{\mathrm{x},\perp}$ or $\mathbf{H}_{\mathrm{x},\angle}$ respectively will result in discontinuous torque commands because of the rank change of the Jacobian between $\mathbf{J}_{\mathrm{xy}}$ and $\mathbf{J}_{\mathrm{x}}$. Interpolating a scalar transition parameter $1 \geq \alpha \geq 0$ (intermediate arrows between the red and blue ones in Fig. 4.1a and b) which is set to one for $\mathbf{H}_{\alpha=1,\perp} = \mathbf{H}_{\alpha=1,\angle} = \mathbf{H}_{\mathrm{xy}}$ and equal to zero for $\mathbf{H}_{\alpha=0,\perp} = \mathbf{H}_{\mathrm{x},\perp}$, $\mathbf{H}_{\alpha=0,\angle} = \mathbf{H}_{\mathrm{x},\angle}$ results in a continuous transition:

$$\mathbf{H}_{\alpha,\perp} = \mathbf{Q}_{1:2} \begin{pmatrix} 1 & 0 \\ 0 & \alpha \end{pmatrix} \mathbf{Q}_{1:2}^T = \mathbf{q}_1\mathbf{q}_1^T + \alpha\, \mathbf{q}_2\mathbf{q}_2^T \tag{4.21}$$

$$\mathbf{H}_{\alpha,\angle} = \mathbf{Q}_{1:2}\mathbf{R} \begin{pmatrix} 1 & 0 \\ 0 & \alpha \end{pmatrix} \mathbf{R}^{+}\mathbf{Q}_{1:2}^T \,, \tag{4.22}$$

and $\mathbf{N}_{\alpha,\perp} = \mathbf{I} - \mathbf{H}_{\alpha,\perp}$, $\mathbf{N}_{\alpha,\angle} = \mathbf{I} - \mathbf{H}_{\alpha,\angle}$. Even though $\mathbf{H}_{\alpha,\perp}$ is not idempotent for $0 < \alpha < 1$, it still fulfills $(\mathbf{H}_{\alpha,\perp})^T = \mathbf{H}_{\alpha,\perp}$. This is not the case when interpolating between oblique projections with $\mathbf{H}_{\alpha,\angle}$.

A joint-space weighting $\mathbf{W}$ can be imposed when projecting $\boldsymbol{\tau}$ onto the range of $\mathbf{J}_{\mathrm{x}}^T$. This allows to interpolate between $\mathbf{H}_{\mathrm{x},\perp}$ and $\mathbf{H}_{\mathrm{x},\angle}$, resulting in different nullspaces indicated by green arrows in Fig. 4.1c. In Fig. 4.1c the projector

$$\mathbf{H}_{\mathrm{x}}^{\mathbf{W}_\beta} = \mathbf{J}_{\mathrm{x}}^T\big(\mathbf{J}_{\mathrm{x}}\mathbf{W}_\beta\,\mathbf{J}_{\mathrm{x}}^T\big)^{-1}\mathbf{J}_{\mathrm{x}}\mathbf{W}_\beta \text{ with } \mathbf{W}_\beta = \begin{pmatrix} \beta & 0 \\ 0 & 1 \end{pmatrix} \tag{4.23}$$

and variable β is chosen. Hence, $\mathbf{H}_{\mathrm{x}}^{\mathbf{W}_\beta}$ projects onto the range of $\mathbf{J}_{\mathrm{x}}^T$ and $\mathbf{W}_\beta$ determines the angle between range space and nullspace. Choosing $\beta = 1$ (red arrow) and $\beta = -0.17$ (blue arrow) is equivalent to the previous orthogonal and oblique projectors, respectively: $\mathbf{H}_{\mathrm{x}}^{\mathbf{W}_{\beta=1}} = \mathbf{H}_{\mathrm{x},\perp}$, $\mathbf{H}_{\mathrm{x}}^{\mathbf{W}_{\beta=-0.17}} = \mathbf{H}_{\mathrm{x},\angle}$ (similarily for the nullspace projectors). Note that $\mathbf{W}$ is not positive definite for $\beta = 0$. This is however not critical because $\mathbf{J}_{\mathrm{x}}\mathbf{W}_\beta\,\mathbf{J}_{\mathrm{x}}^T$ is scalar.

4.3 Priority Adaptation within Stack-of-Tasks

As final contribution in this chapter the classical Stack-of-Tasks (SoT) scheme is extended for shaping operational space tasks and corresponding nullspaces with the proposed approach introduced in Eq. 4.4. In addition, this section proposes *shaped augmented nullspace projections* $\mathbf{N}_i'$ to shift (partially) released degrees of freedom also to the next level: $\mathbf{N}_1' = \mathbf{I}$ and

$$\mathbf{N}_i' = \left[\mathbf{I} - \begin{pmatrix} \mathbf{J}_1 \\ \vdots \\ \mathbf{J}_{i-1} \end{pmatrix}^T \begin{pmatrix} \mathbf{T}_1 & & \\ & \ddots & \\ & & \mathbf{T}_{i-1} \end{pmatrix}^T \left(\begin{pmatrix} \mathbf{J}_1 \\ \vdots \\ \mathbf{J}_{i-1} \end{pmatrix}^{+} \right)^T \right] \text{ for } i > 1 \,. \tag{4.24}$$

Choosing augmented projections within the extended SoT, each task is projected only once, in contrast to successive projections. In this case the idempotence is not utilized within the control scheme in favor of shaped projections. Recall that related works apply non-idempotent matrix operators in multi-level hierarchies as well and ignore the introduced interference. The formulation ensures that any torque vector can be decomposed uniquely as: $\boldsymbol{\tau} = \boldsymbol{\tau}_{\mathbf{N}_1'} + \cdots + \boldsymbol{\tau}_{\mathbf{N}_i'}$.

In the proposed control scheme (Eq. 4.4 with Eq. 4.24), the coupling between tasks depends only on $\mathbf{T}_i$ which can be chosen with QRD or SVD to manage the interference. This is certainly more favorable than a soft prioritization where all tasks interfere with each other. In addition, integrating task-space acceleration shaping derived in Sec. 4.1.8 for weighted projections into the extended SoT allows to define hierarchies with dynamic consistency in a subset of all DOFs.

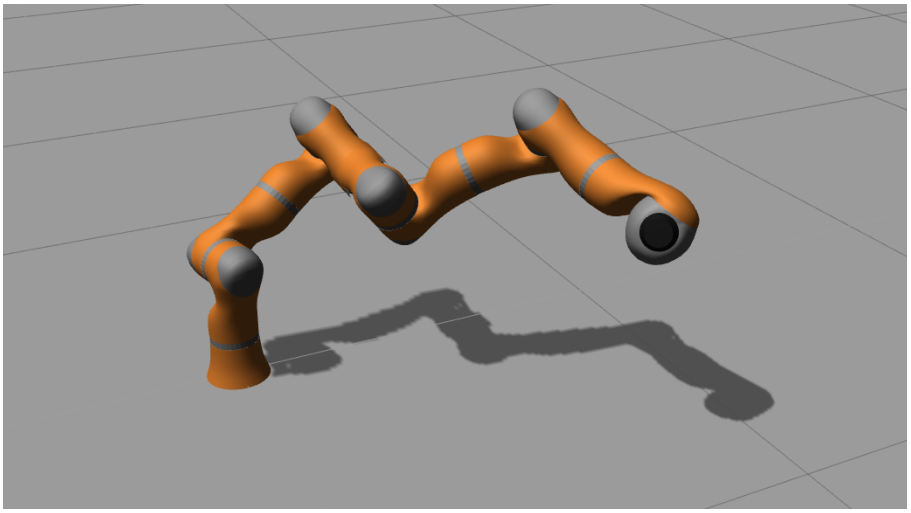


Figure 4.2: Highly redundant artificial 11 DOF manipulator.

When choosing $\mathbf{T}_i = \mathbf{I}$ for all tasks, the proposed formulation reduces to the classical SoT. Note that the computational overhead of the proposed extension is negligible. The analytic solution presented here can readily be applied within typical recursive implementations or quadratic programming.

4.3.1 Simulation

The proposed approach was extensively tested with the dynamics simulation available in Gazebo simulator[1] using ODE. The approach is implemented as analytical control scheme to study general concepts and to evaluate the proposed properties. The experiment is designed such that hardware limits are not violated.

Fig. 4.2 shows the experimental platform, a highly redundant light-weight robot (LWR) with 11 rotational joints inspired by the KUKA LWR manipulator. Three 3D-Cartesian tasks are considered simultaneously: the end-effector as primary task and "elbow", "shoulder" as secondary and third objective. Repetitive circular trajectories in the horizontal plane are chosen with different centers, diameters and frequency for the first two tasks and a static target for the third task. Task conflicts are enforced such that only one task can be achieved successfully.

The experiment is designed as follows: First of all, classical SoT is demonstrated, i.e. all tasks fully activated with $\mathbf{T}_i = \mathbf{I}$ for all i. The inverted inertia matrix is chosen as weighting to obtain a dynamically consistent hierarchy. The next part of the experiment aims for online priority adaptation with the extended SoT (Eq. 4.4 and Eq. 4.24). Single task-space dimensions of the primary and secondary task are linearly shaped while the robot tries to follow the predefined trajectories. The duration of priority adaptations are varied and alternate with a phase of constant priorities as shown in Fig. 4.3. The robot is simulated for a total duration of $70s$.

4.3.2 Results

The desired and estimated horizontal positions of all three operational points are plotted in Fig. 4.4. In the first part of the experiment, the controller tracks the end-effector trajectory precisely while simultaneously minimizing the deviation for the secondary and third task in the respective nullspaces as expected.

After 10 seconds the x-direction of the end-effector task is deactivated (cf. Fig. 4.3), and the released degree of freedom is shifted to the next level in the hierarchy. The end-effector gives

[1] See http://gazebosim.org/

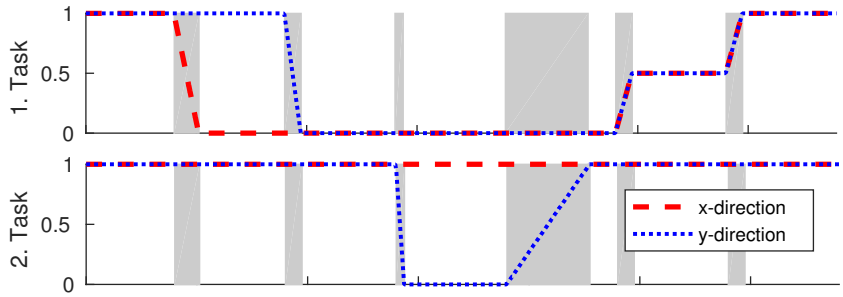


Figure 4.3: Commanded priority deactivation and activation for end-effector (top) and elbow task (bottom). Red indicates the x- and blue the y-direction.

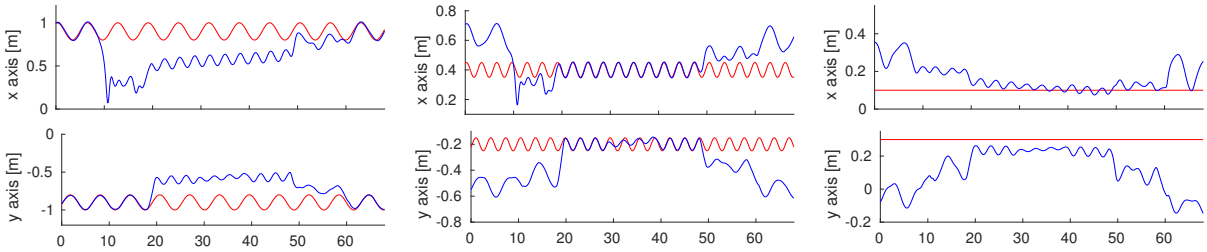


Figure 4.4: Tracking Cartesian tasks with the end-effector (left), "elbow" (middle) and "shoulder" (right). Desired positions are red, estimated are blue. Trajectories for a duration of 70 seconds in the horizontal plane are plotted. Large deviations result from low priorities.

up control in x-direction and the elbow task tracking becomes more precise. Next, successively also the y-direction of the end-effector and elbow task are deactivated. Accordingly, tracking of the third task improves considerably while the x-direction of the elbow task-tracking is not affected.

Lastly, the x- and y-direction of the end-effector are deactivated by 50 percent. As a result, both end-effector and elbow deviate from their desired trajectories which constitutes a compromise as is the case for a soft hierarchy. Finally, the end-effector continues tracking its desired trajectory precisely when fully activated again.

Conclude that the proposed control scheme allows continuous deactivation of single task-space dimensions. The duration of the priority rearrangement can be adjusted according to user requirements.

4.4 Dynamically Consistent Generalized Hierarchical Control

The approach described in the previous section includes aspects from soft prioritization into the dynamically consistent nullspace hierarchy. It enables to continuously insert and delete tasks in the SoT by shaping nullspace projectors, but does not allow to rearrange task priorities. The Generalized Hierarchical Control (GHC) approach instead fully combines strict and soft prioritization into a single prioritization scheme [43,76,77]. Its main advantage is that priorities between each pair of tasks are encoded as scalar values, including the discrete SoT parametrization as a special case. This continuous parametrization allows to swap priorities in a continuous manner. For example, considering two tasks among a set of multiple tasks, the GHC approach enables to transition from a strict higher prioritization via a soft weighted prioritization to a strict lower prioritization. Besides these appealing features, GHC suffers from implementing only orthogonal, statically-consistent nullspace projectors because the inertia matrix is not incorporated as weighting. A detailed analysis and experimental comparison for different SoT implementations in [15] pointed out that a statically-consistent nullspace hierarchy (without

dynamic consistency) may lead to unstable robot motion. The GHC approach however implements only static consistency when parameterized with strict priority relations. To overcome this shortcoming, this section proposes a novel approach referred to as DynGHC which adopts and extends GHC such that dynamic consistency can be achieved while still offering all other advantageous properties.

4.4.1 Continuous Priority Parametrization

For each pair of tasks i, j a separate scalar priority $\alpha_{i,j} \in [0, 1]$ is defined. By convention, task j has a strict lower priority with respect to task i for $\alpha_{i,j} = 0$ and a strict higher priority for $\alpha_{i,j} = 1$. A non-strict, soft priority is achieved for $0 < \alpha_{i,j} < 1$. In this case, task i is partially allowed to move along the directions of task j and becomes stronger restricted for higher values of $\alpha_{i,j}$. A task i is fully deactivated when projecting it onto its own nullspace for $\alpha_{i,i} = 1$, partially activated for $0 < \alpha_{i,i} < 1$ and fully activated for $\alpha_{i,i} = 0$. The entire hierarchy information for all K tasks containing all pairwise priorities is conveniently encoded in matrix form

$$\boldsymbol{\Psi} = \begin{bmatrix} \alpha_{1,1} & & \alpha_{1,K} \\ & \ddots & \\ \alpha_{K,1} & & \alpha_{K,K} \end{bmatrix} \in \mathbb{R}^{K \times K}. \tag{4.25}$$

Continuous priority rearrangement between two tasks i and j is achieved by smoothly varying the value of $\alpha_{i,j}$ and hence also $\alpha_{j,i}$. A existing task i is continuously deleted by regulating $\alpha_{i,i}$ from 0 to 1 and insertion of a new task is accomplished vice versa.

Note that classical strict and soft prioritization schemes are just extreme cases of possible prioritized relations between a set of tasks. Further note that the number of priority-pairs increases with more tasks and the corresponding values are difficult to tune by hand. The continuous parametrization $\boldsymbol{\Psi}$ provides the basis for automated learning priorities, potentially including also time-dependency, analogous to the approach described in chapter 3. This section focuses on the formulation of the DynGHC approach itself and chooses simple priority parametrizations heuristically for experimental validation.

4.4.2 Dynamically consistent shaped Projector

As pointed out earlier in this chapter, shaping projectors enables to project a task completely or partially onto the nullspace of other tasks, or to deactivate it completely. In extension to the statically consistent shaped projector proposed in [43], here a shaped projector with dynamic consistency is proposed. This matrix operator regulates to what extent a lower-priority task is projected onto the dynamically consistent nullspace of a higher-priority task. The final control law synthesizing all K tasks is given by

$$\boldsymbol{\tau} = \sum_{i=1}^{K} \mathbf{N}'_{\mathbf{A}_i} \boldsymbol{\tau}_i \ , \tag{4.26}$$

where the i-th task is represented by the torque vector $\boldsymbol{\tau}_i$ and is multiplied with a shaped projector $\mathbf{N}'_{\mathbf{A}_i} \in \mathbb{R}^{D \times D}$ which depends on the continuous priority parametrization. Alg. 1 computes $\mathbf{N}'_{\mathbf{A}_i}$ and is described in detail in the following. First, a diagonal priority matrix $\mathbf{A}_i$ is required (line 1), containing all pairwise task priorities related to task i (stored in the i-th row of the priority parametrization $\boldsymbol{\Psi}$)

$$\mathbf{A}_i = \begin{bmatrix} \alpha_{i,1}\mathbf{I}_{m_1} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \alpha_{i,K}\mathbf{I}_{m_K} \end{bmatrix}, \tag{4.27}$$

where m_i is the dimensionalality of task i and $\mathbf{I}_{m_i}$ is the $m_i \times m_i$ identity matrix. Next, the diagonal values in $\mathbf{A}_i$ are sorted in descending order resulting in $\mathbf{A}_{i_s}$ (line 4). In addition, the augmented Jacobian (line 1) of all tasks $\mathbf{J}_{aug} = \left[\mathbf{J}_1^T, ..., \mathbf{J}_K^T\right]^T$ is sorted analogously (line 5), leading to $\mathbf{J}_{aug_s}$, such that all rows are ordered according to their influence by task i: tasks that are least influenced appear in the first rows, while tasks that are most influenced appear in the last rows. Extending GHC, a symmetric, positive definite weighting matrix $\mathbf{W} = \mathbf{L}\mathbf{L} \in \mathbb{R}^{D\times D}$ is decomposed employing the Cholesky decomposition where $\mathbf{L} \in \mathbb{R}^{D\times D}$ is a lower triangular matrix (line 6). The product $\mathbf{X} = \mathbf{L}^T\mathbf{J}_{aug_s}^T$ is then factorized according to Alg. 2 (line 7), introducing ϵ as the smallest value greater than zero for numerical reasons. This algorithm is extended from [43]. It performs a QR decomposition (QRD) and skips linear dependent rows of $(\mathbf{J}_{aug_s}\mathbf{L})$ as well as the least important rows in case of $m_1 + ... + m_K > D$. This is the typical situation in multi-task control, e.g. when considering a low important posture task i in joint-space $m_i = D$ besides other operational space tasks. Otherwise, any standard QRD implementation may be used. Alg. 2 returns an orthonormal basis $\mathbf{Q}_i \in \mathbb{R}^{D\times r}$, an upper triangular matrix $\mathbf{R}_i \in \mathbb{R}^{r\times r}$, a permutation vector $\mathbf{p} \in \mathbb{R}^r$ and the rank r. It is important to note that each column in $\mathbf{Q}_i$ is associated to a particular task direction, i.e. rows of $\mathbf{J}_{aug_s}$. Accordingly, the diagonal matrix $\mathbf{A}_{i_s}$ is suited to deactivate such task directions. Next, $\mathbf{A}_{i_s}$ is reduced to a diagonal matrix $\mathbf{A}_{i_{sr}}$ with size $r \times r$ to fit the dimensionality of $\mathbf{Q}_i$. It contains only r diagonal elements of $\mathbf{A}_{i_s}$ corresponding to the selection stored in $\mathbf{p}$ (line 8). Finally, the $\mathbf{W}$-weighted shaped projector $\mathbf{N}_{\mathbf{A}_i}$ for task i is computed as (line 9)

$$\mathbf{N}'_{\mathbf{A}_i} = \left(\mathbf{L}^T\right)^{-1}\left(\mathbf{I} - \mathbf{Q}_i\mathbf{A}_{i_{sr}}\mathbf{Q}_i^T\right)\mathbf{L}^T. \tag{4.28}$$

The derivation is given in the Appendix B. Rearranging priorities is reflected only in an adaptation of $\mathbf{\Psi}$ and $\mathbf{A}_i$, and does not affect the overall computation time. We propose to choose the inverted inertia matrix as weighting $\mathbf{W} = \mathbf{M}^{-1}$ to obtain dynamic consistency for strict priorities as in [11]. Note however that an infinite number of configuration-dependent weighting matrices exist that implement dynamic-consistency [15]. Choosing the identity as weighting $\mathbf{W} = \mathbf{I}$, the DynGHC approach reduces to the basic GHC which features static consistency only. In comparison to the extended SoT described in the previous section, the proposed DynGHC approach incorporates the Jacobians of all tasks in the nullspace computation: the shaped projection $\mathbf{N}_{\mathbf{A}_i}$ for a task i includes also its own Jacobian $\mathbf{J}_i$ and therefore combines projection shaping with operational space task shaping into a single operation. This trick allows to project a task partially or completely onto itself to allow smooth task insertion or removal which was missing in the previous section.

Coupling (or interference) between conflicting tasks has been treated a bit vague before in this chapter. In the following, coupling between two tasks is defined as the acceleration that a torque vector dedicated to a task i introduces in the operational space of another task j:

$$\ddot{\mathbf{x}}_j = \mathbf{J}_j\mathbf{M}^{-1}\boldsymbol{\tau}_i + \dot{\mathbf{J}}_j\dot{\mathbf{q}}\,. \tag{4.29}$$

Recall that task i has null effect on the acceleration of task j if its torque vector is projected onto the dynamically consistent nullspace of task j before, neglecting the effect of $\dot{\mathbf{J}}_j\dot{\mathbf{q}}$:

$$\begin{aligned}\ddot{\mathbf{x}}_j &= \mathbf{J}_j\mathbf{M}^{-1}\left[\mathbf{I} - \mathbf{J}_j^T\left(\mathbf{J}_j\mathbf{M}^{-1}\mathbf{J}_j^T\right)^{-1}\mathbf{J}_j\mathbf{M}^{-1}\right]\boldsymbol{\tau}_i + \dot{\mathbf{J}}_j\dot{\mathbf{q}} = \mathbf{0}\,\boldsymbol{\tau}_i + \dot{\mathbf{J}}_j\dot{\mathbf{q}}\\ \ddot{\mathbf{x}}_j &\approx \mathbf{0}\,.\end{aligned} \tag{4.30}$$

Instead employing the $\mathbf{T}$-shaped projection operator with weighting $\mathbf{W} = \mathbf{M}^{-1}$ discussed in Sec. 4.1.8 yields

$$\begin{aligned}\ddot{\mathbf{x}}_j &= \mathbf{J}_j\mathbf{M}^{-1}\left[\mathbf{I} - \mathbf{J}_j^T\left(\mathbf{J}_j\mathbf{M}^{-1}\mathbf{J}_j^T\right)^{-1}\mathbf{T}\mathbf{J}_j\mathbf{M}^{-1}\right]\boldsymbol{\tau}_i + \dot{\mathbf{J}}_j\dot{\mathbf{q}}\\ \ddot{\mathbf{x}}_j &\approx \left[\mathbf{I} - \mathbf{T}\right]\mathbf{J}_j\mathbf{M}^{-1}\boldsymbol{\tau}_i\,.\end{aligned} \tag{4.31}$$

The interference due to shaping can be interpreted physically on acceleration-level. Consider next projection shaping without weighting ($\mathbf{W} = \mathbf{I}$). The accelerations introduced become

$$\ddot{\mathbf{x}}_j = \mathbf{J}_j \mathbf{M}^{-1} \left[\mathbf{I} - \mathbf{J}_j^T \left(\mathbf{J}_j \mathbf{J}_j^T \right)^{-1} \mathbf{T} \mathbf{J}_j \right] \boldsymbol{\tau}_i + \dot{\mathbf{J}}_j \dot{\mathbf{q}} \tag{4.32}$$

and cannot be interpreted easily.

This analysis highlights that coupling between tasks can occur for two different reasons: both the dynamic model $\mathbf{W}$ and the chosen shaping operator $\mathbf{T}$ affect the accelerations introduced in the operational spaces of other tasks. It is thus clearly beneficial to suppress the undesired inertia coupling between tasks by incorporating an accurate dynamic model within a shaped projection. The GHC approach which does not include the inertia matrix mixes *desired* interferences due to soft priorities with *undesired* interferences due to inertia coupling which can result in unexpected motion. In comparison, the dynGHC which includes the inertia matrix as weighting will stabilize robot control.

4.4.3 Comparative Simulations: Experiment Setups

The proposed DynGHC approach is implemented as an analytical control scheme similar to [15], [21] or [43], but can also be employed in optimization-based control schemes. Note that [77] recently integrated a shaped projector into a quadratic program. Two experiments are conducted in Gazebo simulation environment where accurate physical modeling of the robot is possible.

First, the coupling is evaluated for an artificial, planar 4 DOF manipulator with revolute joints, moving in the y-z-plane subject to gravity. The total mass of the manipulator is $m = 12$ kg and the total length is $l = 2$ m. Each link is $l = 0.5$ m long, has its Center-of-Mass in the

Algorithm 1 Weighted shaped projector for a task i

procedure DYNGHCPROJECTOR($\mathbf{W}, \mathbf{A}_i, \mathbf{J}_{aug_s}, \epsilon$)
  $n \leftarrow GetNbCol(\mathbf{J})$
  $index \leftarrow GetRowsIndexDescOrder(\mathbf{A}_i)$
  $\mathbf{A}_{i_s} \leftarrow SortRows(\mathbf{A}_i, index)$
  $\mathbf{J}_{aug_s} \leftarrow SortRows(\mathbf{J}_{aug}, index)$
  $\mathbf{L} \leftarrow CholeskyDecomposition(\mathbf{W})$
  $\mathbf{Q}_i, \mathbf{R}_i, \mathbf{p}, r \leftarrow modifiedqrd((\mathbf{L}^T \mathbf{J}_{aug_s}^T), \epsilon)$ ▷ Alg. 2
  $\mathbf{A}_{i_{sr}} \leftarrow GetSubDiagMatrix(\mathbf{A}_{i_s}, \mathbf{p})$
  return $\mathbf{N}'_{i,\mathbf{A}_i} \leftarrow \left(\mathbf{L}^T\right)^{-1} \left(\mathbf{I} - \mathbf{Q}_i \mathbf{A}_{i_{sr}} \mathbf{Q}_i^T\right) \mathbf{L}^T$

Algorithm 2 Modified QRD for a matrix $\mathbf{X}$

procedure MODIFIEDQRD($\mathbf{X}, \epsilon$)
  $m \leftarrow GetNbCol(\mathbf{X})$
  $n \leftarrow GetNbRow(\mathbf{X})$
  $i \leftarrow 0$
  for $k \leftarrow 0$ ***to*** $m - 1$ **do**
    if $i \geq n$ **then**
      break
    $\mathbf{Q}[:, i] \leftarrow \mathbf{X}[:, k]$
    for $j \leftarrow 0$ ***to*** $i - 1$ **do**
      $\mathbf{R}[j, i] \leftarrow \mathbf{Q}[:, i]^T \mathbf{X}[:, k]$
      $\mathbf{Q}[:, i] \leftarrow \mathbf{Q}[:, i] - \mathbf{Q}[:, j] \left(\mathbf{Q}[:, i]^T \mathbf{Q}[:, j]\right)$
    $\mathbf{R}[i, i] \leftarrow norm(\mathbf{Q}[:, i])$
    if $\mathbf{R}[i, i] > \epsilon$ **then**
      $\mathbf{Q}[:, i] \leftarrow \mathbf{Q}[:, i] \,/\, \mathbf{R}[i, i]$
      $\mathbf{p}[i] \leftarrow k$
      $i \leftarrow i + 1$
  $r \leftarrow i$
  return $\mathbf{Q}, \mathbf{R}, \mathbf{p}, r$

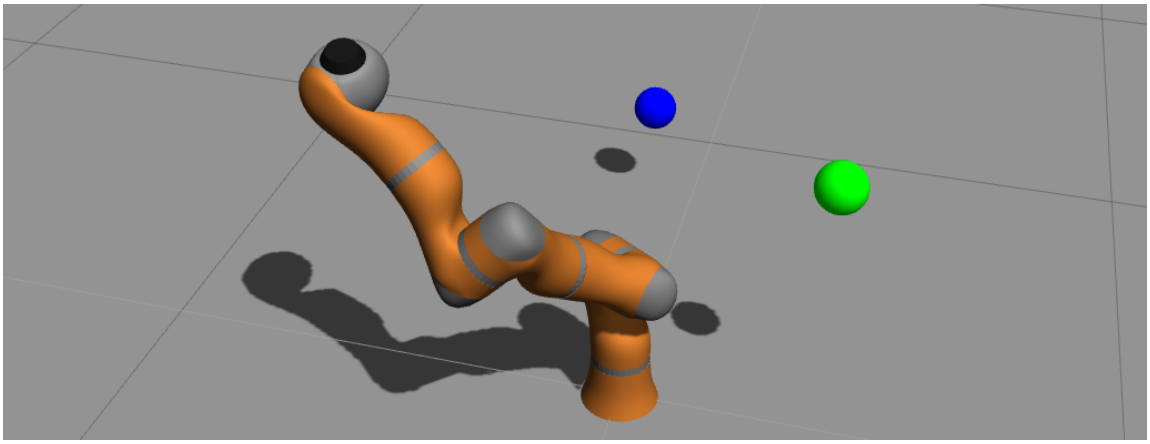


Figure 4.5: Three simultaneous tasks for KUKA LWR: desired joint-space posture (shown) and conflicting positions for end-effector (blue) and elbow (green).

midpoint of the link, weighs $m = 3$ kg and is modeled by a point mass. Two simultaneous tasks A and B are considered: The y and z position of the tip located at the end of the fourth link (A) and the y and z position at the end of the second link (B). Standard PD-control is applied with proportional $\mathbf{K} = 200$ and derivative $\mathbf{D} = 20$ gains for both tasks. Two non-conflicting, static targets are defined, i.e. both tasks can be successfully achieved independently of their relative prioritization: $\mathbf{x}_A^* = (-1.6, 0)^T$ and $\mathbf{x}_B^* = (-0.8, 0)^T$. The manipulator is initialized with $\mathbf{q} = (1.15, 0.85, 0.5, 0.6)^T$rad such that no target is reached at the beginning similar to [15].

The task prioritization $\mathbf{\Psi}$ stays constant during simulation. Both tasks are fully activated ($\alpha_{A,A} = \alpha_{B,B} = 0$) and their relative priority is varied in comparative simulations.

- $\alpha_{A,B} = 0$ and $\alpha_{B,A} = 1$: task A has a strict higher priority with respect to task B (i.e. $A \triangleright B$).
- $\alpha_{A,B} = 0.5$ and $\alpha_{B,A} = 0.5$: task A and B have equal priority (i.e. $A = B$).
- $\alpha_{A,B} = 1$ and $\alpha_{B,A} = 0$: task B has a strict higher priority with respect to task A (i.e. $B \triangleright A$).

In addition, GHC and DynGHC are compared by choosing the respective weighting matrices, resulting in total of six independent simulations. Considering two Cartesian tasks, the control law simplifies to

$$\boldsymbol{\tau}_{CMD} = \mathbf{N}_B^{'}\mathbf{J}_A^T\mathbf{F}_A + \mathbf{N}_A^{'}\mathbf{J}_B^T\mathbf{F}_B \ , \tag{4.33}$$

with $\mathbf{N}_B^{'} = \left[\mathbf{I} - \mathbf{J}_B^T\left(\mathbf{J}_B\mathbf{W}\mathbf{J}_B^T\right)^{-1}\alpha_{A,B}\,\mathbf{J}_B\mathbf{W}\right]$ and $\mathbf{N}_A^{'} = \left[\mathbf{I} - \mathbf{J}_A^T\left(\mathbf{J}_A\mathbf{W}\mathbf{J}_A^T\right)^{-1}\alpha_{B,A}\,\mathbf{J}_A\mathbf{W}\right]$.

The second experiment evaluates priority rearrangement within a hierarchy consisting of three simultaneous tasks for varying transition times. As experimental platforms serves KUKA LWR. For both end-effector (A) and elbow (B) conflicting static Cartesian targets are defined and a constant joint-space posture (C) as the third task which is inconsistent with A and B (cf. Fig. 4.5). At the beginning, the prioritization is parameterized such that all tasks are fully activated. The elbow task has a strict lower priority w.r.t. the end-effector task (i.e. $A \triangleright B$) and a strict higher priority w.r.t. the posture task (i.e. $B \triangleright C$). The initial priority parametrization is encoded as

$$\mathbf{\Psi}_{A \triangleright B \triangleright C} = \begin{bmatrix} \alpha_{A,A} = 0 & \alpha_{A,B} = 0 & \alpha_{A,C} = 0 \\ \alpha_{B,A} = 1 & \alpha_{B,B} = 0 & \alpha_{B,C} = 0 \\ \alpha_{C,A} = 1 & \alpha_{C,B} = 1 & \alpha_{C,C} = 0 \end{bmatrix} . \tag{4.34}$$

Next, the strict prioritization between end-effector and elbow tasks is continuously swapped, resulting in $\alpha_{A,B}=1$,$\alpha_{B,A}=0$ after successful rearrangement (i.e. $B \triangleright A$). During the transition

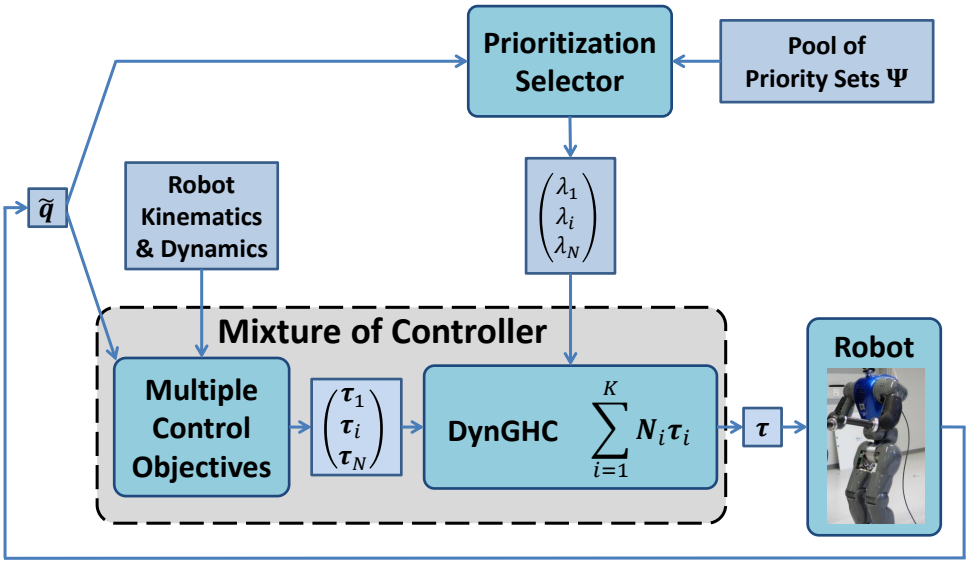


Figure 4.6: Soft and strict prioritization between tasks combined into DynGHC. Lower-part: A multitude of controllers pursue different objectives and compute torques $\boldsymbol{\tau}_i$ which are superimposed employing shaped nullspace projections $\mathbf{N}_i$ related to the prioritization $\boldsymbol{\Psi}$. Upper-part: The prioritization selector is responsible for task-priority rearrangement. It chooses a set of priorities $\boldsymbol{\Psi}$ and can blend between solutions based upon a pool of priorities.

time period $[t_1, t_2]$ the parameterization is smoothly modified employing a sigmoid function analogous to [43]

$$\alpha_{i,j}(t) = 0.5 - 0.5\cos\left(\frac{t - t_1}{t_2 - t_1}\pi\right), \text{ with } t \in [t_1, t_2] \ , \tag{4.35}$$

$$\alpha_{j,i}(t) = 1 - \alpha_{i,j}(t) \ . \tag{4.36}$$

Here the idea illustrated previously for priority rearrangement within soft priorities in Fig. 3.7 can be adopted for DynGHC as shown in Fig. 4.6. The robot is simulated for a total duration of $5.5s$ and the DynGHC approach is compared with the GHC approach as baseline.

4.4.4 Results for Constant Priorities with 4 DOF Manipulator

Fig. 4.7 compares three different relative priorities between tasks A and B with both GHC and DynGHC. The charts in the first two rows show the accelerations that task A introduces in the operational space of task B and vice versa. Within the dynamically-consistent SoT (green) achieved with $\alpha_{A,B} = 0$ (left column) and $\alpha_{A,B} = 1$ (right column) the strict lower prioritized task has null effect on the primary task: $\ddot{\mathbf{x}}_{A\to B} = 0\left[\frac{m}{s^2}\right]$ (first row, right chart), $\ddot{\mathbf{x}}_{B\to A} = 0\left[\frac{m}{s^2}\right]$ (second row, left chart) throughout the entire simulation as intended. The statically-consistent SoT (red) however cannot fully decouple the tasks as expected and in accordance with [15]. Generated accelerations $\ddot{\mathbf{x}}_{A\to B} > 1000\left[\frac{m}{s^2}\right]$, $\ddot{\mathbf{x}}_{B\to A} > 300\left[\frac{m}{s^2}\right]$ in the beginning of the simulation are clearly not desirable. We observe coupling between tasks with both GHC and DynGHC when choosing equal priorities $\alpha_{A,B} = 0.5$ (middle column) which is desired because of the soft prioritization. However, DynGHC performs better compared to GHC due to the additional compensated inertia coupling between tasks.

The charts in the third and fourth row show the accelerations applied by task A and B in their own operational spaces $\ddot{\mathbf{x}}_{A\to A}$, $\ddot{\mathbf{x}}_{B\to B}$. This allows comparison with the scale of disturbance provided in the first two rows of Fig. 4.7: The disturbances due to inertia coupling within GHC are of the same order of magnitude and cannot be neglected.

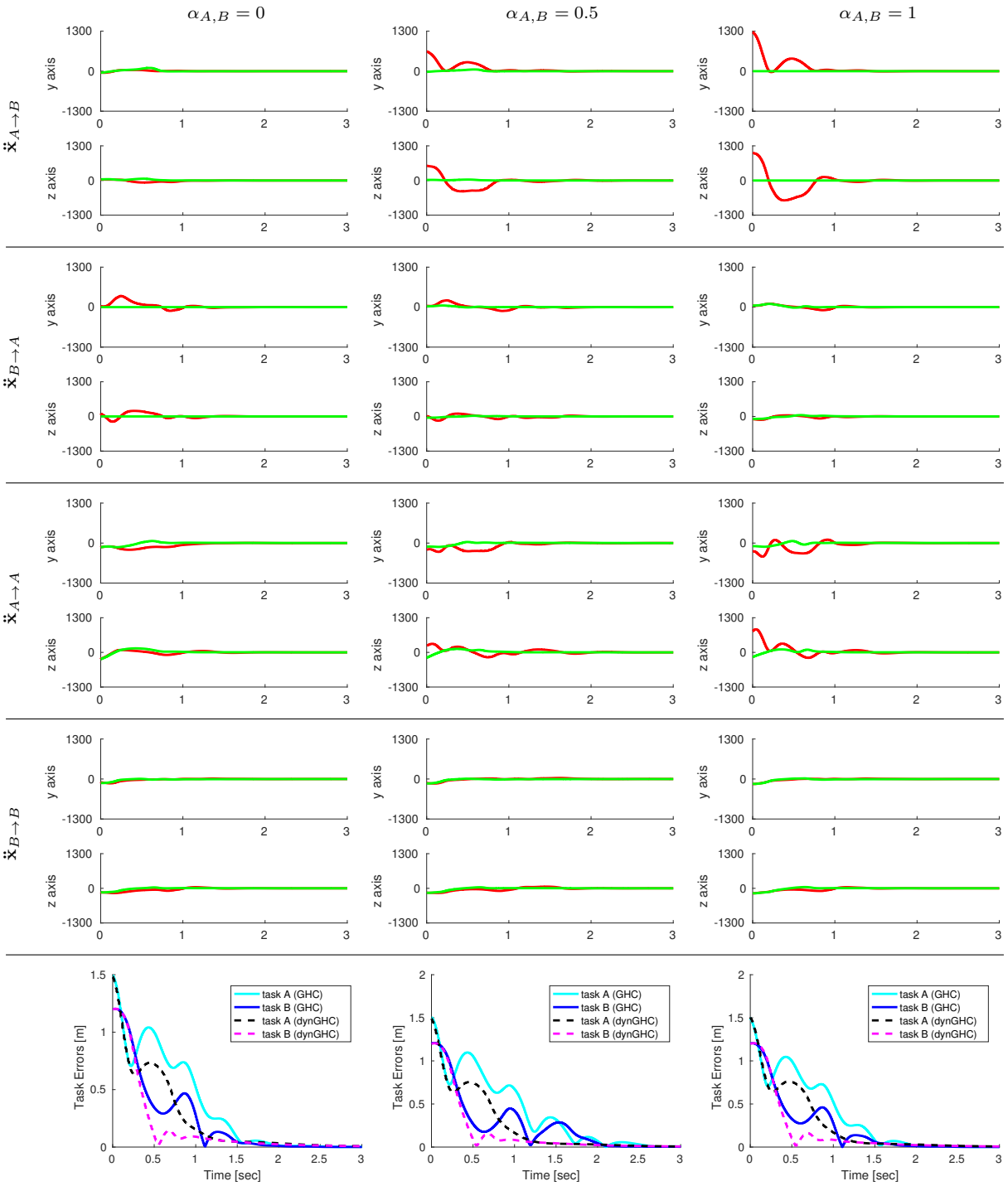


Figure 4.7: Comparative simulations with a 4DOF planar manipulator. The charts are generated with $\alpha_{A,B} = 0$ (left column), $\alpha_{A,B} = 0.5$ (middle column) and $\alpha_{A,B} = 1$ (right column). The four upper rows show accelerations $\ddot{\mathbf{x}}\left[\frac{m}{s^2}\right]$ in the operational space of task A or B, with red indicating classical GHC and green dynamically consistent DynGHC: acceleration introduced by task A into task B (first row), acceleration introduced by task B into task A (second row), acceleration of task A in its own space (third row), acceleration of task B in its own space (fourth row). The bottom row depicts task errors employing the Euclidean norm. All simulations are stopped after three seconds (x-axis in all charts).

The charts in the last row of Fig. 4.7 depict the task errors obtained for all three relative priorities. Note that high errors at the beginning of the experiment are intentional and relate to the initial configuration. The DynGHC approach performs better than GHC in all simulations. With GHC it takes longer to approach steady-state. Furthermore, for $\alpha_{A,B} = 1$ within GHC the robot does not approach steady-state within three seconds, instead, it violates hard joint limits.

4.4.5 Results for Priority Rearrangement with KUKA LWR

Fig. 4.8 first row presents the smooth adaptation of priorities. The second and third row plot the evolution of task errors (utilizing the Euclidean norm) for the KUKA LWR. At the beginning of the experiment, the end-effector reaches its static target precisely with both implementations GHC and DynGHC. In addition, the deviation for the secondary and third task are minimized in the respective nullspaces. The robot is at rest and there are no differences in the initial joint configurations as expected. Both schemes constitute classical SoT with static or dynamic consistency respectively and there is no difference at steady-state. Next, transitions with a total duration of $0.1s$, $1.0s$, and $4.0s$ are initiated. Consequently, the elbow task tracking improves substantially while the end-effector task tracking declines. The robot continues to optimize tasks after the priority transition ended.

The charts in the bottom row of Fig. 4.8 depict the magnitude of joint velocities. The motion generated with DynGHC is much smoother than with GHC. This phenomenon is due to the existence of inertia couplings between tasks. These couplings are implicitly annihilated by nullspace projectors with dynamic consistency. The results show that this feature is also beneficial when shaping dynamically consistent projections, as is the case during priority rearrangement.

Finally, the robot stops moving in all simulations. Zero steady-state error is reached for the elbow task and the end-effector points towards its desired target, locally optimal with respect to the remaining available nullspace. The final joint configurations and consequently also the final task errors are similar for all simulations.

The simulations conducted in this section comparing GHC with DynGHC constitute a strong argument for the use of the inertia matrix as weighting also in non-strict hierarchies with DynGHC.

4.5 Comparison with state-of-the-art

The main drawback of classical SoT is that the prioritization between tasks has to stay constant over time to prevent discontinuities in the control commands. However, smooth insertion and removal of hierarchy-levels is necessary when integrating unilateral, inequality tasks/constraints analytically into the strict hierarchy without requiring quadratic programming. Note that deactivating irrelevant tasks that are not needed enables to forward the remaining redundancy to the next lower priority task [14]. This ability is also required when sequencing different tasks or constraints, for example when establishing a contact. The first attempt to solve this problem is reported in [73] for two strictly prioritized tasks. Following this direction, the approach in [78] enables to deactivate and activate tasks within multiple levels of a strict hierarchy while maintaining the order of priorities.

Continuously rearranging priorities within SoT without deactivating tasks has been studied extensively [78–80]. Related works often restrict swapping operations to consecutive levels in the stack. Many of these solutions are computationally inefficient (especially for many simultaneous tasks in transition) or allow task-insertion and removal only at the lowest priority level.

In an attempt to compare these different approaches, Tab. 4.1 groups related works with respect to five important criteria. Formulating a flexible prioritization scheme that combines all advantages from strict and soft prioritization, requirements include that such scheme

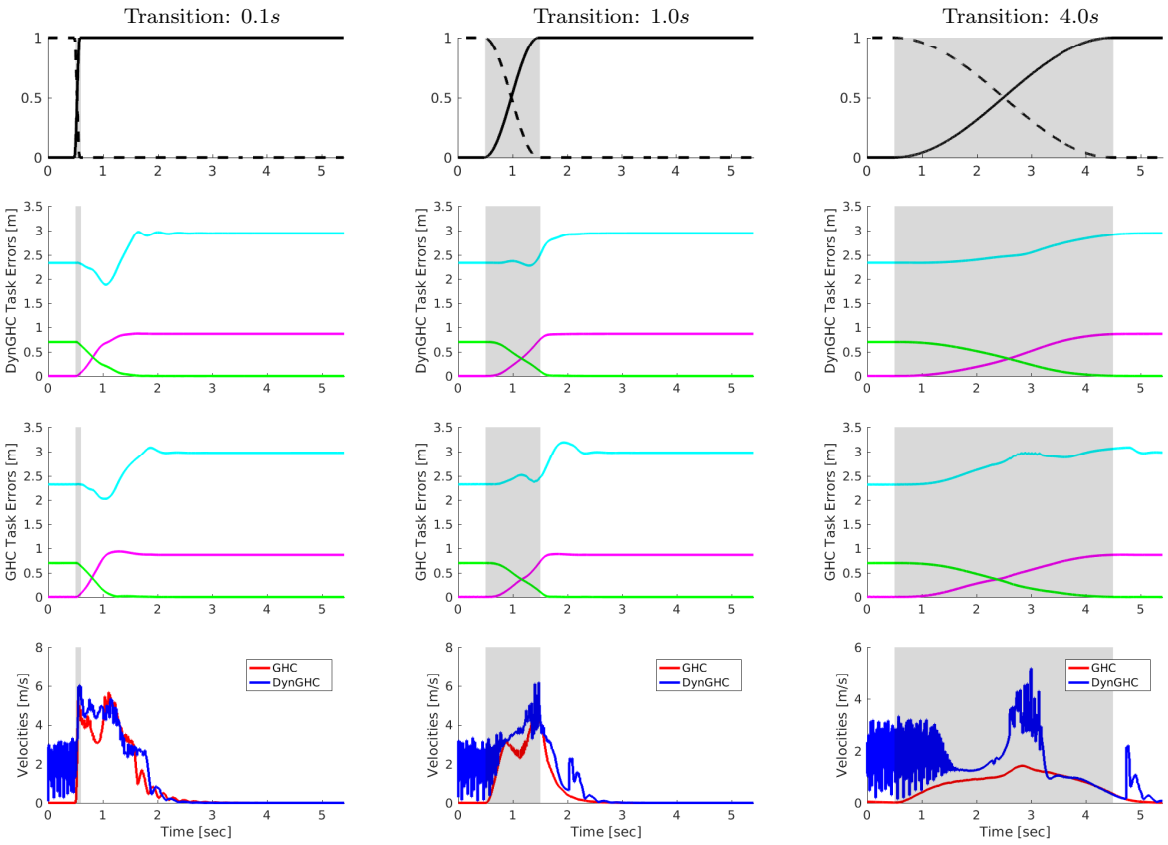


Figure 4.8: Priority rearrangement with KUKA LWR considering three simultaneous tasks. The first row shows the smooth regulation of the priority pair $\alpha_{A,B}(t)$ (solid) and $\alpha_{B,A}(t)$ (dashed). Transition times $0.1s$ (left), $1.0s$ (middle) and $4.0s$ (right) are indicated by shaded areas. Task errors achieved with DynGHC (second row) and GHC (third row) are plotted for end-effector (magenta), elbow (green) and posture (cyan). Large deviations result from low priorities. The last row shows the magnitude of all joint velocities $\|\dot{\mathbf{q}}\|$ within DynGHC (red) and GHC (blue). All simulations are stopped after 5.5 seconds.

R1 provides a continuous priority parametrization that facilitates learning time-dependent priorities and enables to choose a separate prioritization for each pair of tasks, offering two alternatives: (i) a dynamically-consistent strict priority with one task having null effect (zero accelerations) on the other task or (ii) a soft priority with both tasks affecting each other,

R2 allows task insertion on and removal from any priority level,

R3 enables smooth priority rearrangement (i.e. swapping priorities) at runtime while limiting the torque derivative,

R4 allows multiple simultaneous task priority transitions, and

R5 treats many simultaneous task-priorities in transition without computational overhead.

It is important to note that soft weighting schemes discussed in the previous chapter already fulfill aspects **R2-5** by design and are not listed in Tab. 4.1. Further note that GHC [43, 76, 77] satisfies **R1** only partially because the inertia matrix is not incorporated and dynamic consistency is not achieved. The approach in [72] cannot swap arbitrary levels in the hierarchy and cannot provoke a soft prioritization. [81] recently proposed a soft weighting of different strict hierarchies. By regulating the scalar weights associated to each candidate SoT smooth

priority rearrangement is achieved. In this approach, it is however non-intuitive how to choose the scalar weights to obtain a specific prioritization. The authors instead learn time-dependent scalar weights based upon demonstrations. This approach is computationally exhaustive as many SoT hierarchies have to be computed first and it is not easy to see the advantage compared to GHC and DynGHC.

The work on continuous nullspace projection shaping focusing on specific orthonormal bases presented in [21,43] constitutes the basis for the more general approach proposed in this chapter. It is important to stress the fact that the proposed DynGHC performs considerably better compared to the GHC approach. Employing the inertia matrix in the computation of the nullspace enables to design priority parametrizations with dynamic consistency even for non-strict hierarchies.

4.6 Conclusions

Matrix operators which approximate projections are widely employed in robotics. These shaped projections are not idempotent and lead to sub-optimal solutions in the control law which is often ignored. This chapter gives an effective means to shape projections and operational space tasks ensuring that the obtained solutions are still optimal for specified task directions. Furthermore, this chapter shows that classical DLS regularization is a special case of the proposed general formulation. Conversely, projection shaping represents a DLS approach with variable regularization term which is directly interpretable in a geometric fashion – in contrast to previous methods that proposed to adjust the damping factor. Sophisticated damping terms restrict the interference between control objectives. The results obtained constitute a strong argument for the use of shaped projectors which modify only specific directions.

Based on these mathematical foundations, shaping was employed within the strict SoT scheme. The extended SoT approach enables to adapt priorities for single task dimensions continuously without introducing interference with other dimensions. It generates smooth transitions instead of hard switches for insertion and removal of tasks. Finally, the proposed DynGHC approach implements higher flexibility compared to other classical prioritization schemes which are based on either strict or soft priorities. It generalizes dynamically consistent SoT and weighted soft prioritization into a unique prioritization scheme, enabling smooth transitions between both schemes. Within DynGHC, desired coupling between tasks due to soft priorities is isolated from undesired inertia coupling. Furthermore, the ability to rearrange priorities between any subset of tasks simultaneously without computational overhead is a clear advantage compared to alternative approaches. The effectiveness of the approach was demonstrated for a redundant torque-controlled robot tracking multiple tasks simultaneously while rearranging the priorities and limiting the torque derivative. Even though the results are promising, they were experimentally verified only with hierarchies of two and three levels in simulation with artificial but yet informing examples. Further research is necessary to evaluate these approaches with multi-level hierarchies on real robots for useful applications. Additionally, future work is dedicated to proof stability for DynGHC as it has been done for SoT [31] and Weighted Mixture [7]. The analysis provided in this chapter focused on torque-control but application to velocity-control is straightforward.

Most control schemes aiming for priority adaptation presented in this chapter have been applied to fully-actuated robots moving in free space only because dealing with contact constraints is challenging. This observation motivates the next chapter, which focuses on underactuated robots in multi-contact situations.

Table 4.1: Overview of SoT extensions aiming for smooth adaptation of task priorities.

Approaches (ordered w.r.t. year of publication)	**R1** continuous priority parametrization with strict or soft priorities for each pair of tasks	**R2** smooth task insertion and removal	**R3** continuous swapping of tasks priorities	**R4** many simultaneous tasks in transition	**R5** constant computational cost w.r.t. the number of simultaneous tasks in transition
[73] [21] [82] [83]	—	(✓)	—	—	—
[78] [22] [79] [80] [42]	—	(✓)	—	✓	✓
[23] [84]	—	✓	✓	—	—
[85] [86] [41] [87] [88] [89] [90]	—	✓	✓	✓	—
[72] [81]	—	✓	✓	✓	✓
[43] [76] [77]	(✓)	✓	✓	✓	✓
proposed DynGHC	✓	✓	✓	✓	✓

Chapter 5

Decoupling Objectives in Contact

Projected Inverse Dynamics Control (PIDC) is an approach for robots in multi-contact situations that decouples contact wrench control and motion generation. Chapter 2 presented PIDC for the fully-actuated case. When controlling legged robots, however, control schemes have to consider the underactuation which is inherent in the floating-base formulation [91]. [29] and [30] extended PIDC for the underactuated case, employing quadratic programming to project control torques for desired contact wrenches onto the actuated joints. These works combine the inequality and equality constraints imposed by friction cones and underactuation and solve them together. This chapter instead analytically extends the PIDC approach for robots with passive degrees of freedom focusing on the equality constraint imposed by underactuated systems without the need of applying optimization techniques.

First, an illustrative example with a fully-actuated robot wiping a board serves as a basis to demonstrate decoupling of multiple objectives within PIDC. This chapter then focuses on the problem of force-closed grasping a rigid object with multiple manipulators representing fingers. Inspired by virtual model control [92] and the idea of additional virtual contacts in the grasp matrix [93], a multi-arm robot is modeled as an underactuated system, and contacts are employed to resolve underactuation. Treating the object as an additional link virtually attached to the robot allows incorporating the object dynamics into the controller, which is a key feature to perform manipulation tasks accurately [94]. Furthermore, this chapter shows that modeling and controlling floating-base multi-leg robots is equivalent to the proposed approach for multi-arm robots.

A domain analysis presented in [95] identified a number of generic-purpose components which are implemented as reusable building blocks. A large panel of experiments on real robots in this chapter highlight that different scenarios can easily and with little effort be programmed based on such components. Main parts of this chapter have been published in [95–97].

5.1 Illustrative Example demonstrating PIDC

Consider the case of wiping a table with the 7 DOF KUKA LWR and a tool with a flat surface attached to the end-effector as shown in Fig. 5.1. Three different control objectives can be distinguished: (i) applying the desired contact force in the z-direction, (ii) tracking a task-space trajectory with the end-effector in the horizontal x-y-plane and (iii) employing the remaining redundancy to stay close to a desired joint-space posture. In this example, it is obvious that the constrained and unconstrained subspaces are orthogonal to each other.

This wiping scenario previously described in [54] is reproduced here in simulation for validation purpose of the software framework. Fig. 5.2 top depicts the overall control architecture. A detailed domain analysis identified core components which are implemented as reusable C++

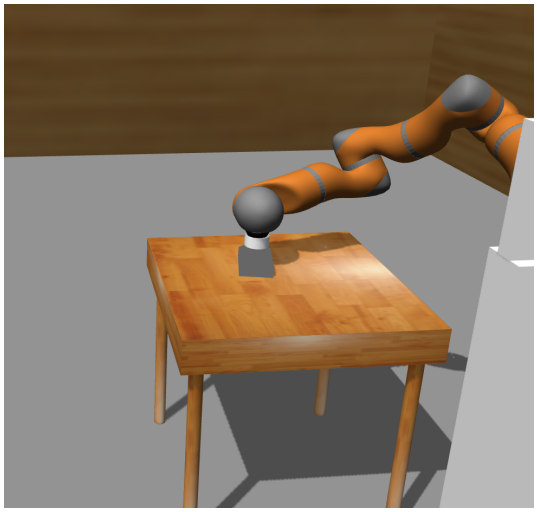


Figure 5.1: The KUKA LWR manipulator wipes a table while generating desired contact forces and simultaneously maintaining a desired joint posture.

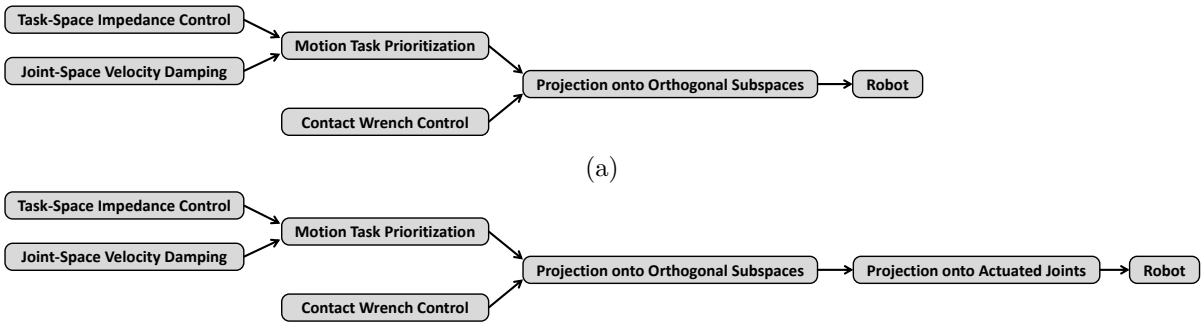


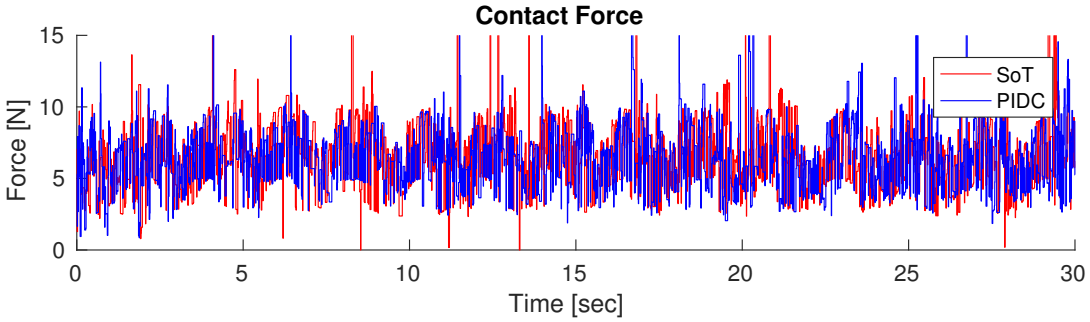
Figure 5.2: Block diagram describing the proposed control architecture for robots subject to contact constraints. Note that the fully-actuated case (a) is a special case of the more general underactuated case (b). A domain analysis revealed core components that are visualized as blocks and implemented separately.

Orocos[1] components. Refer to [95] for more information regarding the domain-specifc-modeling approach.

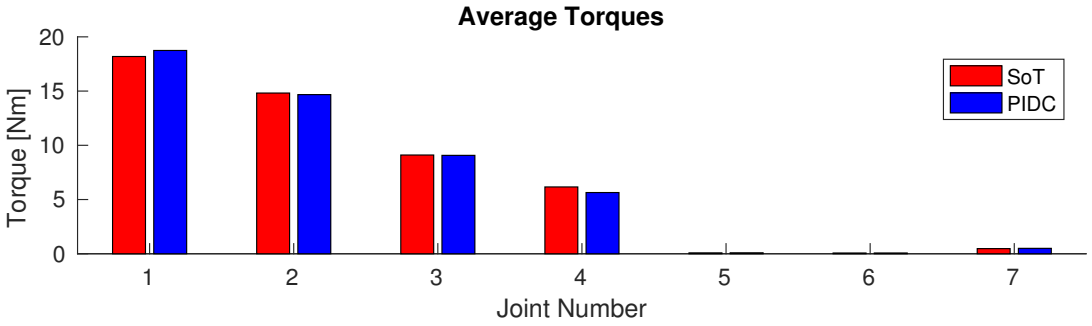
In contrast to [54], here additionally the orientation of the end-effector is controlled and the PIDC approach is compared with the classical Stack-of-Tasks as a baseline. The robot is commanded to apply a force of 5 N to the surface while wiping a circular end-effector trajectory with radius 0.1m as primary motion task and maintaining a desired constant joint configuration as secondary motion task.

Fig. 5.3 shows the resulting contact forces and average joint torques. Even though the PIDC approach requires an orthogonal projection operator without inertia weighting, both approaches achieve similar results w.r.t. tracking precision, achieved contact forces and energy consumption. Therefore the PIDC approach is advantageous: Without the inertia matrix in the nullspace projection PIDC will be more resistant to modeling errors and result in more stable robot control.

[1]See http://www.orocos.org



(a)



(b)

Figure 5.3: Comparison between PIDC and SoT for the table wiping scenario. a) Both approaches manage to fulfill the desired contact force task of 5 N. The contact force sensor simulated in Gazebo provides noisy measurements. b) Absolute average commanded torques to fulfill motion and force control.

5.2 PIDC extended for Underactuated Robots

For a robot system with passive degrees of freedom, the following equality constraint always must be satisfied [28]:

$$\boldsymbol{\tau} = \mathcal{S}\boldsymbol{\tau} \text{ , or equivalently } (\mathbf{I} - \mathcal{S})\,\boldsymbol{\tau} = \mathbf{0} \,, \tag{5.1}$$

with a diagonal matrix $\mathcal{S} \in \mathbb{R}^{D\times D}$ selecting actuated joints.

In the case of decoupled motion and contact wrench control, and assuming that motion control requires all actuated DOFs of the robot, it may still be possible to satisfy Eq. 5.1 by adding constraint wrenches to resolve underactuation, without inducing any additional motion. Following and extending [28], in

$$\boldsymbol{\tau} = \mathbf{N}_c\boldsymbol{\tau}_u + (\mathbf{I} - \mathbf{N}_c)\,\boldsymbol{\tau}_c + (\mathbf{I} - \mathbf{N}_c)\,\boldsymbol{\tau}_s \,, \tag{5.2}$$

$\mathbf{N}_c$ and $(\mathbf{I} - \mathbf{N}_c)$ are the orthogonal projections onto the unconstrained (motion) space and the constrained (wrench) space respectively. Furthermore, $\boldsymbol{\tau}_u$ performs motion control, $\boldsymbol{\tau}_c$ applies desired contact wrenches and $\boldsymbol{\tau}_s$ are torques that induce only contact wrenches to resolve underactuation (Eq. 5.1). Inserting Eq. 5.2 into Eq. 5.1 results in

$$\begin{aligned}(\mathbf{I} - \mathbf{N}_c)\,\boldsymbol{\tau}_s - \mathcal{S}\,(\mathbf{I} - \mathbf{N}_c)\,\boldsymbol{\tau}_s &= \mathcal{S}\,\mathbf{N}_c\boldsymbol{\tau}_u - \mathbf{N}_c\boldsymbol{\tau}_u + \mathcal{S}\,(\mathbf{I} - \mathbf{N}_c)\,\boldsymbol{\tau}_c - (\mathbf{I} - \mathbf{N}_c)\,\boldsymbol{\tau}_c \\ (\mathbf{I} - \mathcal{S})\,(\mathbf{I} - \mathbf{N}_c)\,\boldsymbol{\tau}_s &= (\mathbf{I} - \mathcal{S})\,[-\mathbf{N}_c\boldsymbol{\tau}_u - (\mathbf{I} - \mathbf{N}_c)\,\boldsymbol{\tau}_c]\,. \end{aligned} \tag{5.3}$$

Solving for $\boldsymbol{\tau}_s$ employing the Moore-Penrose inverse to obtain the minimum possible $\|\boldsymbol{\tau}\|$ yields[2]

$$\boldsymbol{\tau}_s = [(\mathbf{I} - \mathcal{S})\,(\mathbf{I} - \mathbf{N}_c)]^{+}\,(\mathbf{I} - \mathcal{S})\,[-\mathbf{N}_c\boldsymbol{\tau}_u - (\mathbf{I} - \mathbf{N}_c)\,\boldsymbol{\tau}_c]\,. \tag{5.4}$$

[2] assuming that all robot actuators are rotary joints

Substituting into Eq. 5.2, provided Eq. 5.4 has at least one valid solution for $\boldsymbol{\tau}_s$, the control equation is written as

$$\boldsymbol{\tau} = \mathbf{N}_c\boldsymbol{\tau}_u + (\mathbf{I}-\mathbf{N}_c)\,\boldsymbol{\tau}_c + (\mathbf{I}-\mathbf{N}_c)\left[(\mathbf{I}-\mathcal{S})\,(\mathbf{I}-\mathbf{N}_c)\right]^{+}(\mathbf{I}-\mathcal{S})\left[-\mathbf{N}_c\boldsymbol{\tau}_u - (\mathbf{I}-\mathbf{N}_c)\,\boldsymbol{\tau}_c\right]. \quad (5.5)$$

Because $(\mathbf{I}-\mathcal{S})$ and $(\mathbf{I}-\mathbf{N}_c)$ are both orthogonal projections, the equation simplifies to[3]

$$\begin{aligned}\boldsymbol{\tau} &= \mathbf{N}_c\boldsymbol{\tau}_u + (\mathbf{I}-\mathbf{N}_c)\,\boldsymbol{\tau}_c + \left[(\mathbf{I}-\mathcal{S})\,(\mathbf{I}-\mathbf{N}_c)\right]^{+}\left[-\mathbf{N}_c\boldsymbol{\tau}_u - (\mathbf{I}-\mathbf{N}_c)\,\boldsymbol{\tau}_c\right]\\ &= \left[\mathbf{I}-\left[(\mathbf{I}-\mathcal{S})\,(\mathbf{I}-\mathbf{N}_c)\right]^{+}\right]\left[\mathbf{N}_c\boldsymbol{\tau}_u + (\mathbf{I}-\mathbf{N}_c)\,\boldsymbol{\tau}_c\right]. \end{aligned} \quad (5.6)$$

The solution can be further simplified

$$\boldsymbol{\tau} = \left[\mathbf{N}_c\mathcal{S}\right]^{+}\mathbf{N}_c\boldsymbol{\tau}_u + \left[\mathbf{I}-\left[(\mathbf{I}-\mathcal{S})\,(\mathbf{I}-\mathbf{N}_c)\right]^{+}\right](\mathbf{I}-\mathbf{N}_c)\,\boldsymbol{\tau}_c\ . \quad (5.7)$$

Note that $\left[\mathbf{N}_c\mathcal{S}\right]^{+}\mathbf{N}_c = \left[\mathbf{Q}_2^T\mathcal{S}\right]^{+}\mathbf{Q}_2^T$ when computing the orthogonal projection onto the nullspace $\mathbf{N}_c = \mathbf{I}-\mathbf{Q}_1\mathbf{Q}_1^T = \mathbf{Q}_2\mathbf{Q}_2^T$ based on a QR decomposition of the Jacobian transpose $\mathbf{J}_c^T = [\mathbf{Q}_1\mathbf{Q}_2]\left[\mathbf{R}^T\mathbf{0}\right]^T$. Consequently, Eq. 5.7 is equivalent to the orthogonal projection approach derived in [98], except that the formulation derived here additionally enables contact wrench control. Further note that Eq. 5.7 reduces to the formulation for a fully-actuated robot system (i.e. no passive joints, $\boldsymbol{\tau}_s = \mathbf{0}$) with $\mathcal{S} = \mathbf{I}$.

5.3 Multi-Limb Robots in Multi-Contact Situations

Based on the above considerations for underactuated system in general, in the following, control of multi-arm and multi-leg robots is specified in more detail. The below considerations describing contact situations build upon the grasp matrix introduced in Sec. 2.3.1.

5.3.1 Accounting for Object Dynamics with Multi-Arm Robots

Previous works on bimanual grasping typically neglect object dynamics and due to that reason, handled only light-weight objects. In contrast, the authors in [36] report an experiment for lifting an object of 12.2kg exhibiting tracking performance issues, due to non-modeled dynamics, that lead to non-zero steady-state error. Next, a formalism to account for object dynamics during grasping is derived.

The external wrench $\mathbf{F}_{ext} \in \mathbb{R}^6$ acting on the object, assuming no human interaction, is given in compact form in the world frame by

$$\mathbf{F}_{ext} = \mathbf{M}_o\ddot{\mathbf{x}}_o + \mathbf{h}_o\ , \quad (5.8)$$

with $\mathbf{M}_o \in \mathbb{R}^{6\times6}$ as the object inertia, $\mathbf{h}_o \in \mathbb{R}^6$ containing the gravitational and Coriolis effects of the motion and $\ddot{\mathbf{x}}_o \in \mathbb{R}^6$ expressing translational and angular accelerations of the object

$$\mathbf{M}_o = \begin{bmatrix} m_o\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \boldsymbol{I}_o \end{bmatrix} \text{ and } \mathbf{h}_o = \begin{bmatrix} m_o\,\boldsymbol{g} \\ \mathbb{S}(\boldsymbol{\omega}_o)\boldsymbol{I}_o\boldsymbol{\omega}_o \end{bmatrix}, \quad (5.9)$$

where $m_o \in \mathbb{R}^D$ is the total mass of the object, $\boldsymbol{I}_o \in \mathbb{R}^{3\times3}$ is the symmetric inertia tensor, $\boldsymbol{\omega}_o \in \mathbb{R}^3$ represents angular velocities of the object, $\boldsymbol{g} = [0, 0, -9.81]^T$ is the gravity vector and choosing the object frame such that it coincides with the object's Center-of-Mass (CoM).

Virtual model control includes virtual components in the control law to move the robot as if these simulated virtual components exist. The external wrench is modeled as a *virtual manipulator* with six DOF acting directly on the object's CoM. Accordingly, the generalized coordinates are then extended with the object pose $\mathbf{x}_o \in \mathbb{R}^6$ containing translation and orientation

$$\mathbf{q} \in \mathbb{R}^{D+6} = \left[\mathbf{q}_1^T, ..., \mathbf{q}_B^T, \mathbf{x}_o^T\right]^T, \quad (5.10)$$

[3] According to [28, Appendix C], for two orthogonal projection operators $\mathbf{X}$ and $\mathbf{Y}$ holds $\mathbf{X}\left[\mathbf{YX}\right]^{+}\mathbf{Y} = \left[\mathbf{YX}\right]^{+}$

a *virtual contact* represented by $\mathbf{G}_o = \mathbf{I}_{6\times6}$ is added to the grasp matrix (Eq. 2.35)

$$\mathbf{G} = [\mathbf{G}_1, ..., \mathbf{G}_B, \mathbf{I}_{6\times6}], \tag{5.11}$$

and equation Eq. 2.37 is accordingly modified to

$$\mathbf{J}_{ee} = \begin{bmatrix} \mathbf{J}_1 & & & \mathbf{0} \\ & \ddots & & \\ & & \mathbf{J}_B & \\ \mathbf{0} & & & \mathbf{I}_{6\times6} \end{bmatrix}. \tag{5.12}$$

Furthermore the robot dynamics given by $\mathbf{M}$ and $\mathbf{h}$ in Eq. 2.38, Eq. 2.39, are extended with the object dynamics $\mathbf{M}_o$ and $\mathbf{h}_o$

$$\mathbf{M} = \begin{bmatrix} \mathbf{M}_1 & & & \mathbf{0} \\ & \ddots & & \\ & & \mathbf{M}_B & \\ \mathbf{0} & & & \mathbf{M}_o \end{bmatrix}, \tag{5.13}$$

and

$$\mathbf{h} = \left[\mathbf{h}_1^T, ..., \mathbf{h}_B^T, \mathbf{h}_o^T\right]^T. \tag{5.14}$$

This formulation can be interpreted as including the object as part of the robotic system as a free-floating link which does not have any actuation, being connected by six virtual joints. This representation allows the robot dynamics to include the dynamics of this additional link. However, the robot system becomes underactuated and cannot be controlled with Eq. 2.49. Instead, the proposed control scheme for the underactuated case Eq. 5.7 can be employed, introducing a diagonal matrix $\mathcal{S} \in \mathbb{R}^{(D+6)\times(D+6)}$ which selects active joints

$$\mathcal{S} = \begin{bmatrix} \mathbf{I}_{D\times D} & \mathbf{0}_{D\times6} \\ \mathbf{0}_{6\times D} & \mathbf{0}_{6\times6} \end{bmatrix}. \tag{5.15}$$

Without knowledge about object dynamics, the object's dynamical parameters can easily be set to $m_o = 0$ and $\boldsymbol{I}_o = \mathbf{0}_{3\times3}$, which then corresponds to the fixed-base PIDC approach[4]. However, it is easy to update the object model at runtime, e.g. during a water-bottle pouring tasks.

A Cartesian impedance behavior can be imposed for the object's Center-of-Mass (CoM) $\mathbf{x}_o \in \mathbb{R}^6$. Extending the classical impedance law described in Sec. 2.2.1 for underactuated PIDC results in the motion control law

$$\boldsymbol{\tau}_u = \mathbf{J}_o^T \mathbf{F}_o + \mathbf{N}_o \boldsymbol{\tau}_0 \ , \tag{5.16}$$

with the object Jacobian $\mathbf{J}_o = (\mathbf{G}^+)^T \mathbf{J}_{ee} \in \mathbb{R}^{6\times D}$ and the wrench $\mathbf{F}_o \in \mathbb{R}^6$

$$\mathbf{F}_o = \boldsymbol{\Lambda}_o \left[\ddot{\mathbf{x}}_{\text{ref}} + \mathbf{J}_o \mathbf{M}_c^{-1} \left(\mathbf{N}_c \mathbf{h} - \dot{\mathbf{N}}_c \dot{\mathbf{q}}\right) - \dot{\mathbf{J}}_o \dot{\mathbf{q}}\right] + \mathbf{K}_{\text{des}} \widetilde{\mathbf{x}}_o + \mathbf{D}_{\text{des}} \dot{\widetilde{\mathbf{x}}}_o \ , \tag{5.17}$$

when choosing damping and stiffness matrices $\mathbf{K}_{\text{des}}$, $\mathbf{D}_{\text{des}}$ and the desired inertia identical to the robot inertia. The inertia matrix in object-space becomes $\boldsymbol{\Lambda}_o \in \mathbb{R}^{6\times6} = (\mathbf{J}_o \mathbf{M}_c^{-1} \mathbf{N}_c \mathbf{J}_o^T)^{-1}$ and the dynamically consistent nullspace projector $\mathbf{N}_o \in \mathbb{R}^{D\times D} = \mathbf{I} - \mathbf{J}_o^T \boldsymbol{\Lambda}_o \mathbf{J}_o \mathbf{M}_c^{-1} \mathbf{N}_c$. Nullspace-drift is prevented in the following experiments by adding a joint velocity damping task $\boldsymbol{\tau}_0 = -\delta \dot{\mathbf{q}}$ with gain $\delta > 0$.

The current object pose can be inferred based on the assumption of a rigid grasp with constant relative transformation between end-effectors and object's center. Hence, the approach does not rely on external sensors, such as vision tracking systems. Note that slipping can be detected by evaluating $\mathbf{J}_c \dot{\mathbf{q}} \neq \mathbf{0}$ to update the grasp matrix if necessary.

[4] In this case the inertia matrix is not positive definite anymore and is not invertible in a direct way. However, the upper-left corner is invertible, which does not contain the object related rows and columns.

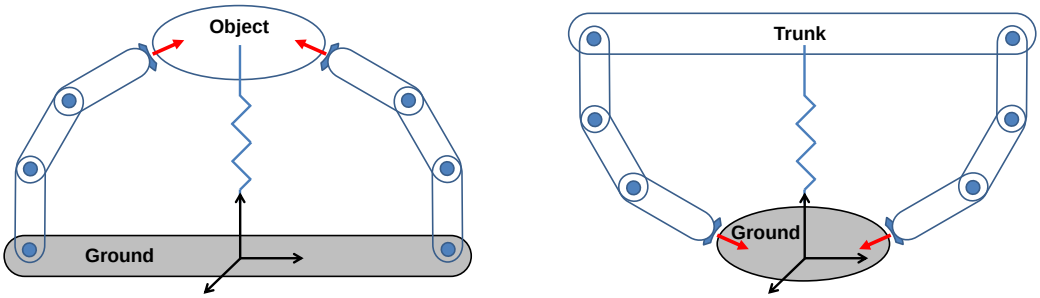


Figure 5.4: Schematic view of underactuated kinematic tree structures for a multi-arm robot rigidly grasping an object modeled with six virtual joints (left) and a multi-leg robot which is connected to the world frame by six virtual DOF (right). The contact constraints and floating-base representations are fundamentally similar.

5.3.2 Relation to Floating-Base Multi-Leg Robots

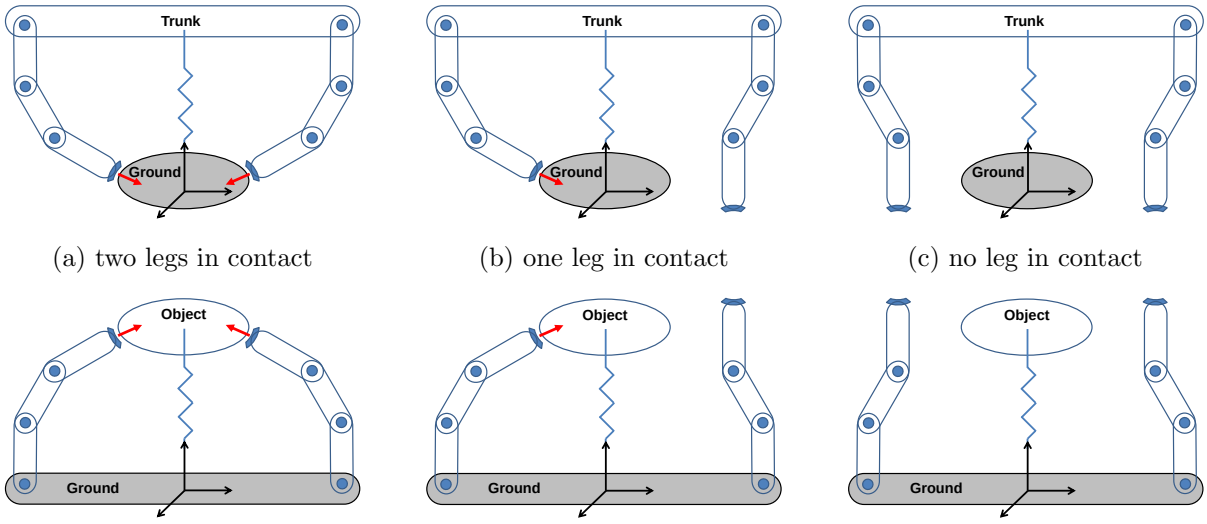
The world inertial frame coincides with the fixed-base of the multi-arm robot when modeling grasping situations. Accordingly, a free-floating object is modeled with six virtual joints, representing the position and orientation of the object's frame with respect to the world inertial frame. However, this is not the only choice: The object frame can also be chosen as world inertial frame, such that the object is rigidly connected to the world. From this perspective, the previously free-floating object becomes static (fixed-base) and the previously fixed-base multi-arm robot turns into a floating-base robot, as illustrated in Fig. 5.4. This kind of setting is typically used for multi-leg robots such as quadrupeds or humanoids, and the object is referred to be the ground. It is noteworthy that the object dynamics in that representation do not play any role: there exist no object dynamics because the object is rigidly connected to the world. On the other hand, dynamics of the robot's base are considered, which were irrelevant before.

Observing this equivalence, the previously derived multi-arm controller can be applied also to multi-leg robot systems. Instead of applying Cartesian impedance for the grasped object, the impedance behavior relates then to the base of the legged robot. Similar to the previous section, maintaining the contact constraint represents a closed kinematic chain for bipedal systems or a closed kinematic tree for robots with more than two legs in contact. The virtual joints representing the robot's floating-base are treated identical to the virtual joints representing the free-floating object in the previous section. This equivalence extends the similarity observed for contact situations of multi-arm and multi-leg robots presented in Fig. 2.2.

Note that both multi-leg and multi-arm robots are subject to switching contact constraints. Fig. 5.5 shows exemplary situations for a biped and a bimanual robot. The main advantage of the floating-base approach for multi-leg robots is that the system can be modeled even when no contacts exist in the environment, e.g. when the robot is in flight or in the air. This also translates to multi-arm robots as objects can be modeled flying in free space as well in a similar fashion. There are then three distinct cases (see also [98] for floating-base multi-leg robots):

1. Multi-arm robots can be seen as fully-actuated when there are as many constraints as free-floating DOFs (e.g. only one arm in contact). The dynamic model can be reduced to a fixed-base model with the object rigidly attached to the end-effector. There is exactly one solution provided that the desired accelerations are constraint-consistent.

2. Multi-arm robots can be underactuated when there are less constraints than free-floating DOFs (e.g. loosing all contacts to the object). In this case full control authority is lost. There is at most one solution to the inverse dynamics problem: for a solution to exist, the desired accelerations must not only be constraint-consistent, they moreover need to be consistent with the dynamics.



(a) two legs in contact (b) one leg in contact (c) no leg in contact

(d) two arms in contact (e) one arm in contact (f) no arm in contact

Figure 5.5: Multi-leg (top) and multi-arm (bottom) robots are subject to switching contact constraints. Underactuation can be resolved by establishing contacts: overactuated case (left), fully-actuated case (middle) and underactuated case (right) assuming flat surface contacts and at least six actuated joints in each robot limb. Both multi-leg and multi-arm robots can be modeled and controlled equivalently.

3. Multi-arm robots can be overactuated when there are more constraints than free-floating DOFs (e.g. two or more end-effectors in contact) and there exists an infinite choice of possible torques to achieve the desired constraint-consistent motion.

Note that both multi-leg and multi-arm robot systems experience discontinuities in the control law when making or breaking contacts and abruptly adapting the corresponding constraint.

5.3.3 Grasping Free-Floating Objects with Floating-Base Robots

The previous two subsections describe how to model and control (i) a fixed-base multi-arm robot grasping a free-floating object and (ii) a floating-base multi-leg robot grasping a fixed-base object (ground). It is possible to generalize these two approaches.

Consider a situation with a quadruped jumping in the air while still holding an object between its feet. The object dynamics will affect the quadruped. In this situation neither the object nor the robot's base are rigidly attached to the world. Instead, the system dynamics can be modeled by representing both the quadruped's base and the object with each six virtual joints. Note that the formulation derived above for the underactuated PIDC is generic with respect to the number of passive joints and can be employed for such multi-limb robot systems as well. Even though the robot will be highly underactuated during the jump (with less constraints than free-floating DOFs) it is still advantageous being able to model this situation as a multi-limb robot. It is easy to see that the underactuated multi-arm and multi-leg models are specific cases of such a system.

5.4 Grasping Experiments with four-fingered Robot Hand

In the following three separate experiments with a robot system consisting of four manipulators representing a four-fingered robot hand are described. The experiments focus on the analytical control solution solving underactuation. The approach was extensively tested with the dynamics simulation available in Gazebo simulator before conducting experiments

on real robots, reproducing the results. Here results obtained from the real robots only are reported. A supplementary video for agile and dexterous object manipulation with a bimanual robot is online available at `https://youtu.be/KZSfKo8h37E` and with four manipulators at `https://youtu.be/Ao-0W9chAd4`. To the authors knowledge, [97] is the first publication treating more than three industrial manipulators in a cooperative manner for dexterous object manipulation via a force-closed grasp.

5.4.1 Robot Hand Platform

Four KUKA LWR IV+ manipulators composing a highly redundant robot system are mounted on a table (cf. Fig. 5.6). This system is treated as a single underactuated robot with 28 active joints and six virtual joints for the free-floating object. The end-effectors consist of a triangular metal plate with three small rubber feet mounted near each corner. This enables to make stable contact also on non-planar surfaces, even though only a single contact point per manipulator is controlled. Employing the well-known grasp matrix constraint described previously this robot system represents an enormous hand with four fingers, each of approximately 1.2m length when including the end-effectors. The contact forces are heuristically chosen such that they point towards the center of all contacts and contact torques are not applied.

Throughout the grasping experiments two different solid objects are grasped, both with 0.3m height. A cylinder (mass $m = 3.0$kg, radius $r = 0.15$m) made out of transparent plastic and a bulky box (mass $m = 9.2$kg) with an octaedal base area (each surface width $l = 0.2$m) made out of hard plastic. These two objects are chosen to demonstrate manipulation of heavy objects and grasping non-planar surfaces. For simplification, it is assumed that object dynamics as well as object shape with predefined desired contact points and associated contact surface normals are known to the controller. This is legitimate as other works on grasping are based on the same assumptions.

The four manipulators are provided by different labs and vary with respect to their hardware parameters: They have been used for different applications for different amounts of time, also some joints were replaced by the manufacturer. Accordingly, the results showed uneven wear and tear for each unit with different joint friction behavior. However, the current implementation is based on a single dynamic model for all four manipulators.

The final reference torques are sent to the four control units via the KUKA FRI employing the joint-space impedance control mode using the hard real-time Orocos execution environment with Linux and Xenomai as in [43].

5.4.2 Incorporating Object Dynamics during Manipulation

First, the underactuation PIDC approach is demonstrated with the 9.2kg box object. A circular reference trajectory in the horizontal plane is tracked with standard PD-control in Cartesian space and no impedance behavior, Fig. 5.7 shows the resulting tracking performance. All four robot arms cooperatively hold the object without slipping effects while following the desired trajectory precisely.

5.4.3 Online Adaptation according to varying Object Dynamics

The second experiment aims for online adaptation of object dynamics, starting with the 3kg cylinder is at rest. A human adds 3kg sand as illustrated in Fig. 5.8, to drastically double the object's total mass. Accordingly, the cylinder moves down due to the non-compensated additional weight which is treated as an external force. Next, the internal object model is updated manually and consequently the controller moves the object back to its desired position. Results are plotted in Fig. 5.9 and reveal precision issues due to high joint friction with the real robot, which is not the case in simulation.

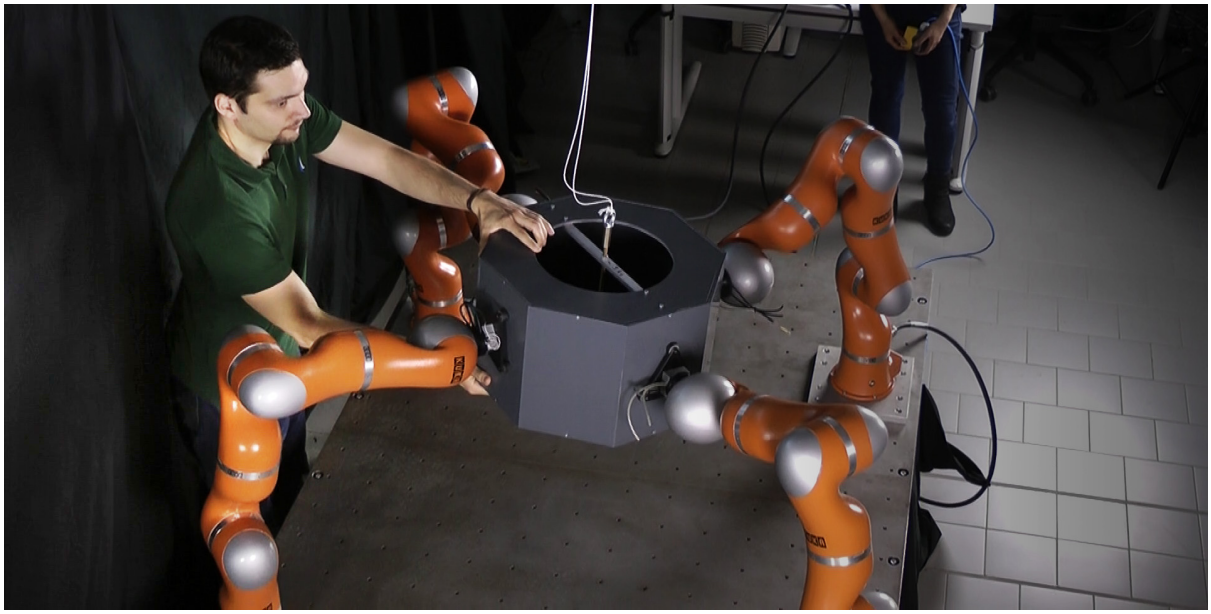


Figure 5.6: An enormous robot hand with four fingers manipulates a 9.2kg object. Compensating for object dynamics enables to provide an impedance-based human-robot interaction mode.

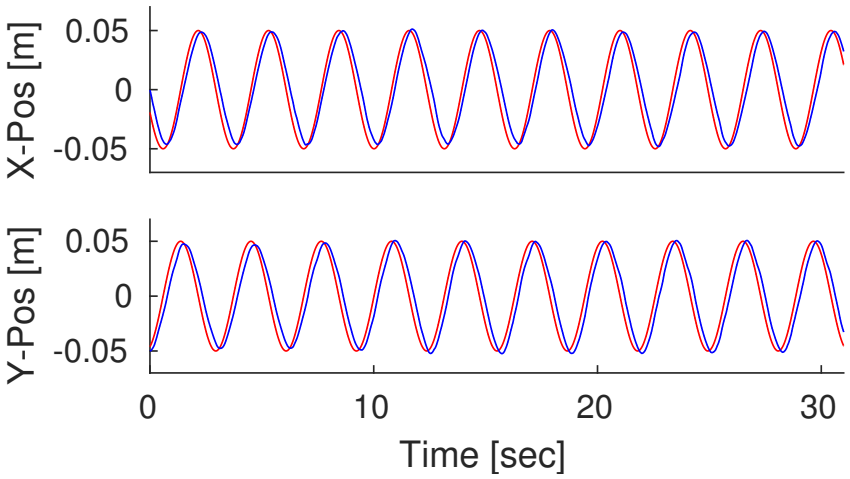


Figure 5.7: Controlling the four-fingered robot hand to accurately track a circular trajectory with a diameter of 0.1 m. The red line indicates the desired object position and the blue line the estimated position.

5.4.4 Assisted Gravity Compensation Mode for Interaction

The Cartesian impedance formulation derived in Eq. 5.16 treats the object as a spring-damper system, transforming deviations from the desired pose into wrenches. This enables a chosen explicit behavior of the system with respect to an external disturbance, e.g. from human interaction with the object. Compensating for object dynamics allows to set the proportional feedback gain to zero (no stiffness) and command zero desired velocities and accelerations to obtain an *assisted gravity compensation mode* with velocity damping. This enables the human to easily move the cumbersome object to a desired position and orientation, where the controller then performs pure gravity compensation. Fig. 5.6 shows a snapshot from the video. Such a mode is highly beneficial in programming-by-demonstration applications, e.g. [4,99] for teaching bimanual manipulation.

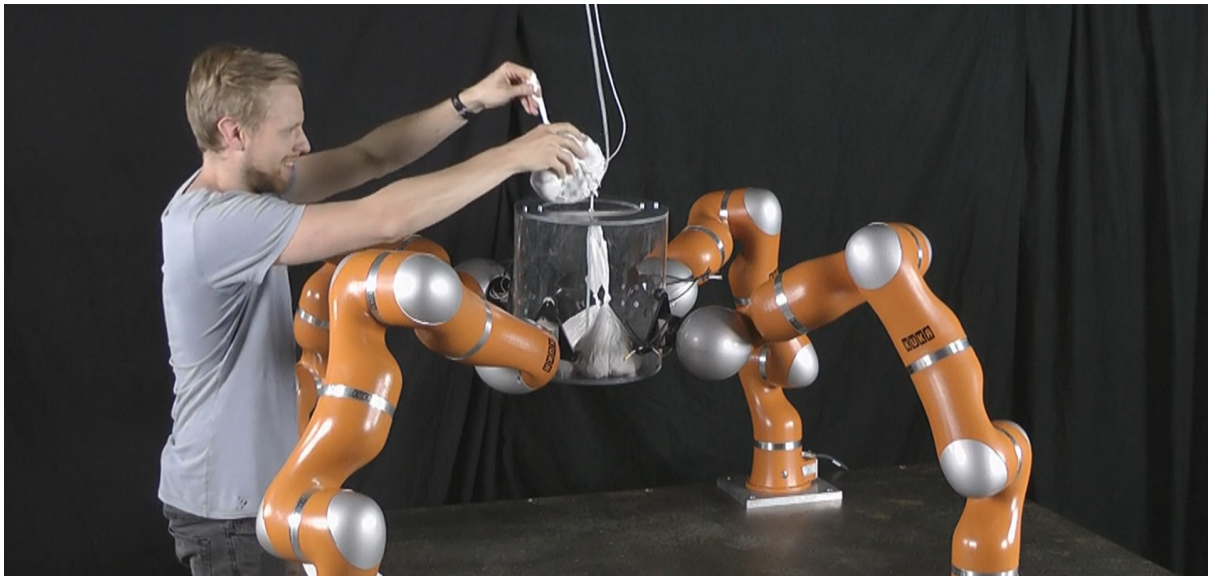


Figure 5.8: When adding mass to the object, the internal model can easily be updated. The controller then compensates for the new object dynamics.

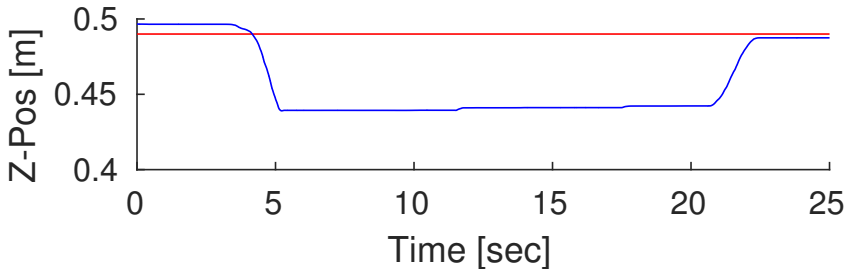


Figure 5.9: The grasped cylinder-like object drops down when adding 3kg mass, resulting in a total weight of 6kg. Next, the internal dynamics model is incrementally updated in three steps, each of adding 1kg to the object dynamic parameters. Because of high joint friction, the object moves back to its initial pose when fully compensating for its total weight. The red line indicates the desired object position and the blue line the estimated position.

5.5 Experiments with underactuated Quadruped

Next, experiments with a quadruped are described. The supplementary video showing all results is available at `https://youtu.be/Ao-0W9chAd4`.

5.5.1 Quadruped Robot Platform

The experimental platform is the ANYbotics quadruped robot, ANYmal (cf. Fig. 5.10), a torque-controlled robot with 12 actuated joints [100]. A dynamic model is delivered by the manufacturer. As is common practice [101], the robot's CoM is approximated with the CoM of the torso. The position and orientation of the robot torso are chosen as the operational space for the Cartesian impedance behavior. The robot feet are treated as point contacts where the positions are fixed on the ground and orientations are not considered. Accordingly, contact moments cannot be controlled. The desired contact forces are set appropriately in order to satisfy the friction cones.

Note that the constraint Jacobian based on the grasp matrix constrains the end-effectors relatively Eq. 2.36. Instead, it is possible to impose an absolute constraint by choosing $\mathbf{J}_c = \mathbf{J}_{ee}$ which is supposed to be more robust to external disturbances.

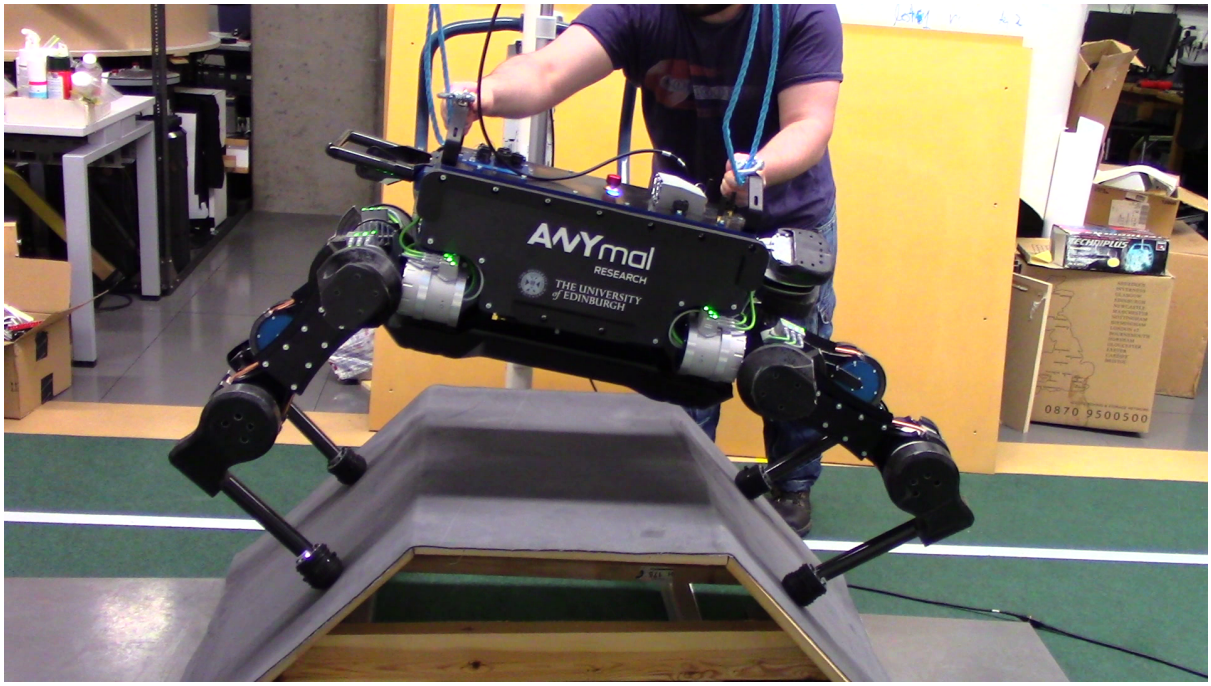


Figure 5.10: ANYmal standing on an a slope ramp. Because of the imposed impedance behavior, the robot reacts to human interaction as a mechanical mass-spring-damper system.

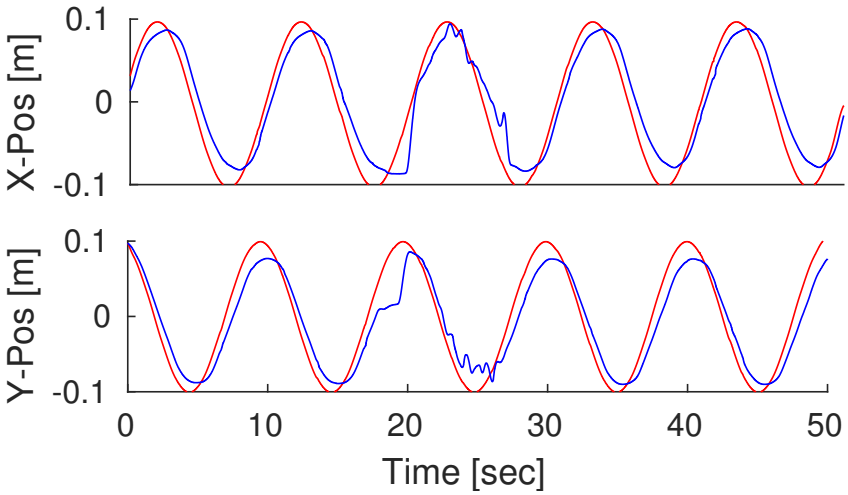


Figure 5.11: The torso of the quadruped tracks a circular trajectory in the horizontal plane. Desired positions are red and estimated blue. After 18 seconds, the robot is perturbed for a short period.

5.5.2 Decoupled Motion and Contact Force Control with ANYmal

In the first experiment with the quadruped, the controller performs on flat terrain demonstrating the robot's workspace. Similar to the previous section, the torso is commanded to follow a circular trajectory inside the horizontal x-y plane, while the end-effectors push toward the ground to avoid slipping. Fig. 5.11 plots the desired and estimated horizontal position of the torso. The trajectory is tracked precisely, the controller also recovers from external disturbances.

5.5.3 Human Interaction while standing on a Ramp with ANYmal

The next experiment demonstrates the robot reactions to human interaction. The robot stands on a slope ramp with inclined contact surfaces of 45 degrees each. In order to show the properties of the Cartesian impedance controller, stiffness and damping are set such that the robot is compliant in x-direction but stiffer in the y- and z-axis. When pushing against the torso, the robot behaves as a mass-spring-damper system while maintaining the contacts. Fig. 5.10 shows a snapshot of the video.

Commanding zero contact forces in simulation the robot fails to maintain contact on the ramp as expected. This shows the underlying advantage of the proposed controller compared to [28, 53, 98] which perform pure motion control with legged robots on flat terrain.

5.6 Comparison with state-of-the-art

Major research efforts have been spent in order to derive projection operators which solve the equality constraints related to underactuation and contact constraints within PIDC. Tab. 5.1 summarizes solutions for different combinations. Note that the most complex case (bottom right cell) has not been properly addressed before and is derived in this chapter. The projection derived generalizes previous works as can be clearly seen by choosing $\mathcal{S} = \mathbf{I}$ and/or $\mathbf{N}_c = \mathbf{I}$.

Early works on underactuated humanoids did not consider the control of contact wrenches [28,53,98]. This was possible because these robots always were placed on flat, horizontal ground. The contacts are coplanar in that specific situation and the grasp matrix degenerates [102]. Contact wrench control is not necessary because gravity enforces contact. The proposed formulation in this chapter for underactuated contact wrench control generalizes [28, 53, 98], enabling to control contact wrenches as well.

[29] and [30] solve the equality constraint imposed by underactuation Eq. 5.1 for controlling contact wrenches based on quadratic programming. The approach described in this chapter instead solves it analytically. The formulation is different compared to dynamically consistent, underactuated SoT [17, 36, 46, 51, 103] because motion control and contact wrench control are projected onto orthogonal subspaces which satisfy the underactuation constraint.

In grasping scenarios, tracking a desired trajectory with the object will be less accurate when neglecting the object dynamics. Manipulating heavy objects is not possible without high PD-feedback gains to correct for the external wrenches, which is not desired. As an illustrative example, consider statically holding an object of 9.2kg with the impedance scheme derived in Eq. 5.16, assuming a perfect dynamics model. The external force due to gravity in the z-direction is $-9.81\frac{\mathrm{m}}{\mathrm{s}^2}9.2\mathrm{kg} \approx -90\mathrm{N}$ and is treated as external disturbance by the impedance controller. Consequently, the proportional gain has to be set to 9000 to achieve tracking accuracy of 0.01m in the z-direction. Without incorporating object dynamics in the control scheme, zero steady-state error cannot be achieved. Furthermore, when increasing the object's weight incrementally, the object will always move down due to non-compensated gravity.

Table 5.1: Summary of analytical projections solving equality constraints related to underactuation (Eq. 5.1) and contacts (Eq. 2.32) within PIDC.

	free space	contact situation: $\mathbf{J}_c\,\ddot{\mathbf{q}} + \dot{\mathbf{J}}_c\,\dot{\mathbf{q}} = \mathbf{0}$	
	motion generation	contact-consistent motion generation	motion generation and contact wrench control
fully-actuated robot	$\boldsymbol{\tau} = \boldsymbol{\tau}_u$	$\boldsymbol{\tau} = \mathbf{N}_c\boldsymbol{\tau}_u$	$\boldsymbol{\tau} = \mathbf{N}_c\boldsymbol{\tau}_u + (\mathbf{I} - \mathbf{N}_c)\,\boldsymbol{\tau}_c$
underactuated robot: $(\mathbf{I} - \mathcal{S})\,\boldsymbol{\tau} = \mathbf{0}$	$\boldsymbol{\tau} = \mathcal{S}\boldsymbol{\tau}_u$	$\boldsymbol{\tau} = (\mathbf{N}_c\,\mathcal{S})^+\,\mathbf{N}_c\boldsymbol{\tau}_u$	$\boldsymbol{\tau} = \left[\mathbf{I} - [(\mathbf{I} - \mathcal{S})\,(\mathbf{I} - \mathbf{N}_c)]^+\right] \cdot [\mathbf{N}_c\boldsymbol{\tau}_u + (\mathbf{I} - \mathbf{N}_c)\,\boldsymbol{\tau}_c]$

To avoid this issue, alternatively, the dynamic model of the end-effector link can be updated as proposed in [104], treating the grasped object as mass-less. However, this strategy is less general and disadvantageous when switching contact constraints. Recently, [94] proposed an approach for incorporating object dynamics in the control law, utilizing the grasp matrix Eq. 2.35. This method, however, is based on a decentralized scheme, controlling each robot arm independently without explicit communication between manipulators and requires a fixed-base dynamic model. [105] demonstrates the ability to add object dynamics for fixed-base robots, requiring wrist-mounted force-torque sensors, which are not needed within the proposed approach in this chapter. Another approach to compensate for object dynamics is to add the term $\mathbf{J}_o^T \mathbf{F}_{ext}$ to Eq. 5.16. However, this causes the corresponding nullspace for a secondary motion task to lose dynamic consistency because the object inertia is missing in the system's joint-space inertia matrix.

Grasping situations are typically treated as fixed-base multi-arm robot systems without virtual joints, e.g. [49, 105–108]. Treating the object as an additional link virtually attached to the robot turns the multi-arm robot into an underactuated system and allows to incorporate the object dynamics into the controller. Furthermore, considering inhand manipulation, this formulation is advantageous compared to previous works on grasping where control schemes have to be switched in case of contact transitions.

The recently proposed work [9] on multi-robot control builds on top of the same idea of including free-floating objects into a single multi-body system. It further extends the approach including articulated mechanisms which are part of the environment, for example, rotational door joints or translational drawer joints. The approach has been validated in a real-world scenario in [109].

The analogy between multi-arm and multi-leg robots has been recognized long ago, however, the similarities when describing the multi-contact situations have been more recent [36, 51, 52, 110, 111]. In both scenarios, the robot tries to achieve a desired wrench via the contact points while the relative transformation between contacts stays constant. This chapter additionally highlights the equivalence between both robot systems under the same modeling and control scheme.

5.7 Conclusions

The PIDC approach for robots in contact situation decouples motion generation and contact wrench control, resulting in different and independent control laws. This chapter analytically extends PIDC to the underactuated case. The unconstrained controller accomplishes several motion tasks independently of grasp forces, e.g. Cartesian trajectory tracking with desired impedance behavior to deal with external disturbances while regulating a secondary joint-space objective in the nullspace. Simultaneously, the constrained component enforces the contact, without affecting the motion tasks. Note that the existence of an analytical solution is beneficial for evaluating the precision of alternative optimization-based approaches. Furthermore, the projection operator for underactuation may be included into the DynGHC scheme – proposed in the previous chapter for fully-actuated robots – as extension to floating-base robots.

The approach presented in this chapter extends earlier work on dexterous grasping by simultaneously torque-controlling four industrial manipulators, representing an enormous robot hand. Object dynamics are included for precise manipulation by employing a virtual manipulator which acts upon the object's Center-of-Mass. Accordingly, the multi-arm robot consisting of fixed-base manipulators is described as an underactuated system. Next, the equivalence between floating-base multi-leg and underactuated multi-arm platforms is highlighted. The present work is then evaluated on both a four-fingered robot hand and a quadruped, demonstrating the controller's robustness and ability to maintain a grasp, subject to unknown human interactions.

In the real-robot experiments described above friction cones are not considered for simplicity. These experiments focus instead on the analytical control solution solving underactuation. Heuristics are employed to verify that the chosen wrenches are sufficient enough to maintain contact to simplify matters, because inequality constraints cannot be be solved analytically by projection operations. In [96], I contributed to optimization of contact wrenches for the fully-actuated PIDC. This wrench optimization approach has been recently extended to the underactuated case [112] based on the analytical projector described here.

Chapter 6

Conclusions

In this thesis, I focus on flexible prioritization among multiple control objectives for highly-redundant robot systems. Classical prioritization schemes, namely strict and soft prioritization, allow for synthesizing several objectives and have been studied extensively during the last decades, however, there remained a number of key challenges: At first, it was unclear how to devise optimal priorities among various control objectives. Priorities were typically tuned by hand based on heuristics. Next, both soft and strict prioritization schemes comprise contrary advantages. A new approach generalizing both schemes and enabling to switch between different prioritization concepts continuously was highly desirable. Finally, employing PIDC for underactuated robots subject to contact constraints, control objectives could only be decoupled based on quadratic programming techniques, and an analytical solution to this problem was missing. Addressing these three main research questions, in this thesis, I specifically

1. developed a framework for automated learning soft task priorities in chapter 3,
2. proposed a generalized prioritization scheme with dynamic consistency in chapter 4, which enables to continuously switch between soft and strict hierarchies, and
3. derived an analytic solution to decouple contact wrench control from motion generation for underactuated robots in chapter 5.

The main contributions of each chapter are briefly summarized in the following.

Chapter 3: Learning Soft Priorities

This chapter studies soft prioritization between tasks with scalar weights representing priorities as an alternative approach to the strict SoT scheme. Priorities have been selected in previous works based on heuristics or prior knowledge. For this reason, a general and repeatable method for automatic tuning priorities from scratch is highly beneficial. Learning techniques are the means to find a suitable set of priorities for dedicated high-level goals. The main contribution of this chapter is the development of a framework which allows for automatic derivation of suitable weights in the torque mixture, representing priorities. It is further proposed to evaluate several variations of the desired goal in each roll-out, in order to achieve robust generalization of the optimized priority set. Additionally, the advantage of regularization is investigated when learning new priorities. The functionality of the optimization framework is demonstrated for a virtual 3 DOF manipulator and the humanoid robot COMAN.

Chapter 4: Managing Interference between Tasks

Both soft and strict prioritization schemes provide oppositional advantages and disadvantages. The strict SoT scheme decouples conflicting control objectives which is beneficial, but hierarchi-

cal priorities have to be devised based on prior knowledge, and continuous priority rearrangement is difficult. In contrast, within soft prioritization control objectives interfere with each other but scalar priorities can be learned offline and rearranging priorities online is straightforward. First, this chapter provides a formal analysis of continuous projection shaping and operational space task shaping. These shaping techniques are the basis to regulate coupling between objectives. A novel prioritization approach is devised which allows to insert new or to remove existing control objectives smoothly. The proposed DynGHC scheme enables continuous priority rearrangement between each pair of tasks, which enables to swap tasks that are not necessarily on consecutive hierarchy levels. Furthermore, the formulation allows to switch between dynamically consistent strict nullspace hierarchies and a soft weighted mixture between objectives while compensating for inertia-coupling between conflicting tasks. The ability to rearrange tasks is a key aspect to deal with contact transitions or changes in the environment, or to react to modified high-level goals.

Chapter 5: Decoupling Objectives in Contact

Controlling underactuated robots subject to contact constraints provides additional complexity compared to motion generation in free space with fully-actuated fixed-base manipulators. Resolving underactuation is elementary to control quadrupeds and humanoids: legged robots are typically represented with a floating-base consisting of six virtual joints describing the position and orientation of the torso. This chapter introduces an analytical solution to decouple contact wrench control from motion generation for underactuated robots in contact situation, resulting in different and independent control laws. A projection operator is derived which solves the equality constraint imposed by underactuation analytically. With this new formulation, the PIDC approach is further extended to the general case of robot systems with passive or virtual joints, generalizing earlier approaches for pure motion generation of underactuated robot systems. The proposed approach is experimentally verified with a large panel of experiments demonstrating dexterous interaction tasks on two different real robot platforms: Four-fingered grasping experiments are performed with an enormous robot hand consisting of four KUKA LWR, cooperatively manipulating heavy objects and a quadruped served as the experimental platform to demonstrate the same grasping controller on a floating-base robot standing on a slope ramp. By treating the grasped object as an additional robot link, which does not have any actuation, the object dynamics are incorporated in the control law resulting in better tracking performance. Moreover, the multi-arm robot becomes underactuated and is modeled and controlled equivalently to a floating-base legged robot which is a key insight in the field of multi-limb robots. Note that the proposed approach is generic for legged robots and robot hands with an arbitrary number of limbs in contact. The experiments are implemented with reusable Orocos components based on a detailed domain analysis, which allows specifying new scenarios with little effort.

Future Directions

The main contributions achieved in this thesis are strongly related to each other but not yet integrated in a common approach. Several avenues of potentially fruitful further research could be based upon combining the key results in this dissertation (cf. Fig. 6.1).

- First, the idea of so-called matrix priorities and the continuous priority parameterization that allow for smooth priority adaptation over time as discussed in chapter 4 opens up prospects for automated tuning more advanced priorities compared to state-of-the-art. Optimization frameworks as proposed in chapter 3 will allow to make the most out of high-dimensional and difficult-to-tune generalized priority sets.

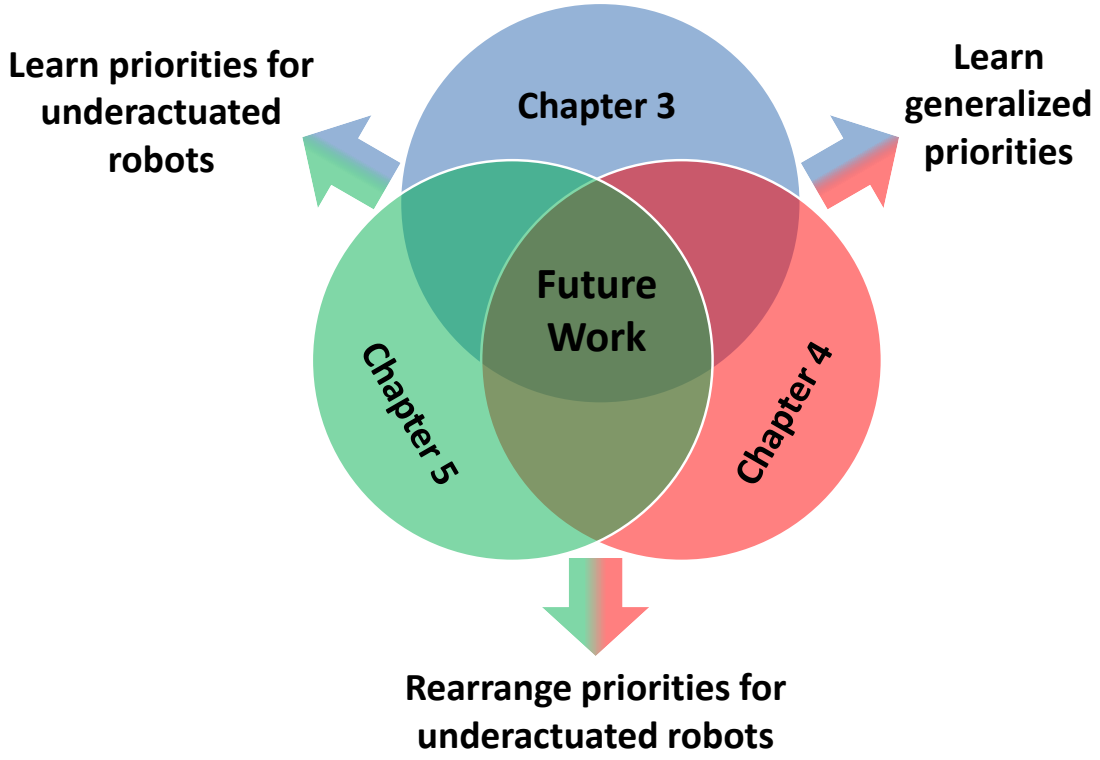


Figure 6.1: Future research could be productively focused on integrating the three main contributions achieved in this thesis in a common framework.

- Another possible improvement worth investigating is how to fuse projection shaping elaborated in chapter 4 with the specific projector solving underactuation derived in chapter 5. Adding these two concepts will enable to apply priority adaptation also for floating-base robots with changing contact situations.
- Finally, merging the two previous steps seems possible and will result in a sophisticated framework for learning a mixture of time-dependent strict and soft priorities applied to underactuated robots.

These advances in follow-up work might potentially extend the boundaries of this thesis toward the real world application of highly redundant robots in challenging environments.

Appendix A

Shaped DLS-Regularization

First classical damped least squares regularization Eq. 2.31 is derived from a SVD-based shaped projection Eq. 4.14 and then its equivalence to a QRD-based projector Eq. 4.15 is shown.

$$
\begin{aligned}
&\mathbf{I} - \mathbf{V}\mathbf{A}_{\mathrm{SVD}}\mathbf{V}^T = \mathbf{I} - \mathbf{V}\left(\left(\mathbf{S}\,{}^{1}\!/_{z^2}\,\mathbf{S}^T\right)^{-1} + \mathbf{I}\right)^{-1}\mathbf{V}^T \\
&= \mathbf{I} - \mathbf{V}\mathbf{S}\mathbf{U}^T\left(\mathbf{U}\mathbf{S}^T\left(\left(\mathbf{S}\,{}^{1}\!/_{z^2}\,\mathbf{S}^T\right)^{-1} + \mathbf{I}\right)\mathbf{S}\mathbf{U}^T\right)^{-1}\mathbf{U}\mathbf{S}^T\mathbf{V}^T \\
&= \mathbf{I} - \mathbf{V}\mathbf{S}\mathbf{U}^T\left(\mathbf{U}\mathbf{S}^T\mathbf{V}^T\mathbf{V}\mathbf{S}\mathbf{U}^T + \mathbf{U}z^2\mathbf{U}^T\right)^{-1}\mathbf{U}\mathbf{S}^T\mathbf{V}^T \\
&= \mathbf{N}_{\mathrm{DLS}} = \mathbf{I} - \mathbf{J}^T\left(\mathbf{J}\mathbf{J}^T + z^2\,\mathbf{I}\right)^{-1}\mathbf{J} \\
&= \mathbf{I} - \mathbf{Q}\mathbf{R}\mathbf{P}^T\left(\mathbf{P}\mathbf{R}^T\mathbf{Q}^T\mathbf{Q}\mathbf{R}\mathbf{P}^T + \mathbf{P}z^2\mathbf{P}^T\right)^{-1}\mathbf{P}\mathbf{R}^T\mathbf{Q}^T \\
&= \mathbf{I} - \mathbf{Q}\mathbf{R}\mathbf{P}^T\left(\mathbf{P}\mathbf{R}^T\left(\left(\mathbf{R}\,{}^{1}\!/_{z^2}\,\mathbf{R}^T\right)^{-1} + \mathbf{I}\right)\mathbf{R}\mathbf{P}^T\right)^{-1}\mathbf{P}\mathbf{R}^T\mathbf{Q}^T \\
&= \mathbf{I} - \mathbf{Q}\left(\left(\mathbf{R}\,{}^{1}\!/_{z^2}\,\mathbf{R}^T\right)^{-1} + \mathbf{I}\right)^{-1}\mathbf{Q}^T = \mathbf{I} - \mathbf{Q}\mathbf{A}_{\mathrm{QRD}}\mathbf{Q}^T
\end{aligned}
$$

Appendix B

Weighted Pseudoinvese & Projector

Consider a symmetric, positive definite weighting matrix $\mathbf{W} \in \mathbb{R}^{D\times D}$ which is decomposed employing the Cholesky decomposition $\mathbf{W} = \mathbf{L}\mathbf{L}^T$, where $\mathbf{L} \in \mathbb{R}^{D\times D}$ is a lower triangular matrix. The $\mathbf{W}$-weighted pseudoinvese of $\mathbf{J} \in \mathbb{R}^{H\times D}$ defined in Eq. 2.4 can be written as

$$\mathbf{J}^{\mathbf{W},+} = \mathbf{W}\mathbf{J}^T\left(\mathbf{J}\mathbf{W}\mathbf{J}^T\right)^{-1} = \mathbf{L}\mathbf{L}^T\mathbf{J}^T\left(\mathbf{J}\mathbf{L}\mathbf{L}^T\mathbf{J}^T\right)^{-1} = \mathbf{L}\left(\mathbf{J}\mathbf{L}\right)^+ \tag{B.1}$$

where the superscript indicates the weighting matrix. Consequently, the weighted pseudoinverse can be computed with the methods derived for the Moore-Penrose inverse without weighting. Hence, with SVD for $\mathbf{J}\mathbf{L} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ and with QRD for $\mathbf{J}\mathbf{L} = \mathbf{P}\mathbf{R}^T\mathbf{Q}^T$ yields

$$\mathbf{J}^{\mathbf{W},+} = \mathbf{L}\,\mathbf{V}\mathbf{S}^+\mathbf{U}^T \text{ and } \mathbf{J}^{\mathbf{W},+} = \mathbf{L}\,\mathbf{Q}\left(\mathbf{R}^T\right)^+\mathbf{P}^T \tag{B.2}$$

The weighted (oblique, non-orthogonal) idempotent projection matrix $\mathbf{H}^{\mathbf{W}}$ onto the range of $\mathbf{J}^T$ (with $\mathbf{H}^{\mathbf{W}}\mathbf{J}^T = \mathbf{J}^T$) defined in Eq. 2.23, becomes, when left-multiplying with $\left(\mathbf{L}^T\right)^{-1}\mathbf{L}^T = \mathbf{I}$

$$\mathbf{H}^{\mathbf{W}} = \mathbf{J}^T\left(\mathbf{J}^{\mathbf{W},+}\right)^T = \mathbf{J}^T\left(\mathbf{J}\mathbf{W}\mathbf{J}^T\right)^{-1}\mathbf{J}\mathbf{W} = \left(\mathbf{L}^T\right)^{-1}(\mathbf{J}\mathbf{L})^T\left((\mathbf{J}\mathbf{L})^+\right)^T\mathbf{L}^T \tag{B.3}$$

With this trick, the weighted projection onto the range of $\mathbf{J}^T$ is derived only with the formulation for an orthogonal projector. The inverse operator is avoided by employing SVD for $\mathbf{J}\mathbf{L} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ or QRD for $\mathbf{J}\mathbf{L} = \mathbf{P}\mathbf{R}^T\mathbf{Q}^T$

$$\mathbf{H}^{\mathbf{W}} = \left(\mathbf{L}^T\right)^{-1}\left(\mathbf{V}\mathbf{V}^T\right)\mathbf{L}^T \text{ and } \mathbf{H}^{\mathbf{W}} = \left(\mathbf{L}^T\right)^{-1}\left(\mathbf{Q}\mathbf{Q}^T\right)\mathbf{L}^T \tag{B.4}$$

The complementary weighted nullspace of $\mathbf{J}^T$ is obtained by $\mathbf{N}^{\mathbf{W}} = \mathbf{I} - \mathbf{H}^{\mathbf{W}}$.

Note that all equations derived in this appendix reduce to their non-weighted equivalents when setting $\mathbf{W} = \mathbf{L} = \mathbf{I}$.

Related References by the Author

This thesis builds on the findings of five conference papers and one journal article, published on major international congresses on robotics research. The following list summarizes all peer-reviewed publications of the author during his doctoral studies, ordered with respect to year of publication. In addition, two workshop contributions have been published [113, 114].

Currently, one additional conference paper describing the proposed DynGHC approach is under review for the *International Conference on Robotics and Automation, 2019*. Furthermore, a workshop proposal "Continuous Management and Scheduling of multiple simultaneous prioritized Tasks for redundant Robots" has been submitted for ICRA 2019 together with Abderrahmane Kheddar and Alin Albu-Schäffer. Four IEEE-RAS Technical Committees have acknowledged their full support of the proposed workshop and top-experts worldwide agreed to present and discuss their perspectives in Montreal, Canada.

[55] Niels Dehio, R. Felix Reinhart, and Jochen J. Steil. Multiple Task Optimization with a Mixture of Controllers for Motion Generation. In IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pages 6416–6421, 2015.

This paper first proposes a framework which automates tuning of task priorities for the soft prioritization approach. The main idea is to treat the scalar weights in the torque superposition as parameters of a policy and to apply standard stochastic policy search (CMA-ES) for policy improvement. Manual tuning of priorities is thus obsolete. This work contributes to chapter 3.

[56] Niels Dehio, R. Felix Reinhart, and Jochen J. Steil. Continuous Task-Priority Rearrangement during Motion Execution with a Mixture of Torque Controllers. In IEEE/RAS Int. Conf. on Humanoid Robots, pages 264–270, 2016.

The main contribution of this paper is the idea to additionally incorporate regularization when learning new soft priorities, extending the approach introduced in [55]. Optimization of priorities is experimentally evaluated for the upper body of the humanoid robot COMAN for two different high-level goals. Furthermore, continuous rearrangement of soft priorities is demonstrated, enabling the robot to react to environmental changes. The results of this work are reproduced in the presented thesis in chapter 3.

[95] Dennis Leroy Wigand, Arne Nordmann, Niels Dehio, Michael Mistry, and Sebastian Wrede. Domain-Specific Language Modularization Scheme Applied to a Multi-Arm Robotics Use-Case. Journal of Software Engineering for Robotics, 8(1):45–64, 2017.

This journal publication presents the requirements for a domain-specific language modularization and composition scheme from a robotics perspective. I mostly contributed to the multi-arm robotics use-case which serves as a running example throughout the paper. The concepts introduced served as guidelines for the decomposition of the "Projected Inverse Dynamics Control" scheme into core components. Such reusable software components simplify implementation of new scenarios with robots subject to contact constraints. Hence, this article is related to chapter 5.

[96] Hsiu-Chin Lin, Joshua Smith, Keyhan Kouhkiloui Babarahmati, Niels Dehio, and Michael Mistry. A Projected Inverse Dynamics Approach for Multi-arm Cartesian Impedance Control. In IEEE/RSJ Int. Conf. on Robotics and Automation, 2018.

This paper employs "Projected Inverse Dynamics Control" for grasping a rigid object with two robot arms. The so-called grasp matrix, originally proposed for a multi-fingered robot hand, is utilized to constrain the bimanual robot. Furthermore, a quadratic program optimizes contact wrenches which satisfy friction cone constraints and minimize reference torques. I contributed to the implementation and extensive testing of the bimanual robot experiment with the aim to extract, verify and realize the concepts described in [95]. The main idea of representing a bimanual robot as a multi-fingered robot hand is explained as part of related works in chapter 2.

[97] Niels Dehio, Joshua Smith, Dennis Leroy Wigand, Guiyang Xin, Hsiu-Chin Lin, Jochen J. Steil, and Michael Mistry. Modeling & Control of Multi-Arm and Multi-Leg Robots: Compensating for Object Dynamics during Grasping. In IEEE/RSJ Int. Conf. on Robotics and Automation, 2018.

This paper analytically extends the "Projected Inverse Dynamics Control" approach for robots in multi-contact situations to the underactuated case. A matrix operator is derived which projects reference torques onto the actuated joints. This enables to compensate for object dynamics in a grasping setup with general multi-arm robots. Furthermore, it is highlighted that modeling and control of legged robots are fundamentally similar. Real robot experiments with a floating-base quadruped and four KUKA LWR representing an enormous robot hand validate the approach. Large parts of this work are reproduced in chapter 5.

[71] Niels Dehio, Daniel Kubus, and Jochen J. Steil. Continuously Shaping Projections and Operational Space Tasks. In IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, 2018.

This work lays the foundations of chapter 4. First, shaping of idempotent matrix operators is mathematically defined and a solution to smooth out discontinuities resulting from rank changes of an orthonormal basis is provided. This paper also shows that damped least squares is a special case of the proposed more general formulation. Finally, the Stack-of-Tasks prioritization scheme is extended for continuous priority rearrangement of single task dimensions.

List of Acronyms

CMA-ES	Covariance Matrix Adaptation Evolution Strategy
COMAN	COMpliant huMANoid
CoM	Center-of-Mass
DLS	Damped Least Squares
DOF	Degree of Freedom
DynGHC	dynamically consistent Generalized Hierarchical Control
GHC	Generalized Hierarchical Control
LWR	light-weight robot
PD	proportional-derivative
PIDC	Projected Inverse Dynamics Control
QP	Quadratic Program
QRD	QR Decomposition
SoT	Stack-of-Tasks
SVD	Singular Value Decomposition

List of Symbols

Throughout this thesis the following conventions are adopted: Matrices, are denoted by upper-case letters, vectors are usually denoted by lower-case boldface letters and scalars are denoted by normal letters. The table below lists symbols used throughout this thesis.

$\mathbf{q}$	joint angles
$\dot{\mathbf{q}}$	joint velocities
$\ddot{\mathbf{q}}$	joint accelerations
$\boldsymbol{\tau}$	joint torques
$\boldsymbol{\tau}_u$	unconstrained joint torques dedicated motion generation
$\boldsymbol{\tau}_c$	constrained joint torques dedicated to contact wrench control
$\boldsymbol{\tau}_s$	constrained joint torques dedicated to resolve underactuation
$\mathbf{x}$	task-space position
$\dot{\mathbf{x}}$	task-space velocity
$\ddot{\mathbf{x}}$	task-space acceleration
$\mathbf{F}$	task-space wrench
$\mathbf{F}_c$	constrained contact wrench
$\mathbf{J}$	Jacobian
$\mathbf{J}_c$	constrained Jacobian
$\mathbf{H}$	projector onto the rangespace
$\mathbf{N}$	projector onto the nullspace
$\mathbf{N}_c$	orthogonal projector onto the nullspace of the contact constraint
$\mathbf{0}$	null matrix
$\mathbf{I}$	identity matrix
$\mathbf{W}$	weighting matrix
$\mathbf{Z}$	damped least squares regularization
$\mathbf{h}$	gravity and Coriolis components in robot dynamics
$\mathbf{M}$	joint-space inertia matrix
$\mathbf{M}_c$	constrained joint-space inertia matrix
$\boldsymbol{\Lambda}$	task-space inertia matrix
$\boldsymbol{\Lambda}_c$	constrained task-space inertia matrix
m	mass
$\mathbf{V}$	orthonormal basis obtained from SVD
$\mathbf{S}$	diagonal matrix obtained from SVD
$\mathbf{U}$	orthonormal matrix obtained from SVD
$\mathbf{Q}$	orthonormal basis obtained from QRD
$\mathbf{R}$	upper triangular matrix obtained from QRD
$\mathbf{P}$	permutation matrix obtained from QRD

r	rank
$\mathcal{S}$	selection matrix for actuated joints
$\mathcal{R}$	rotation matrix
$\mathbf{T}$	transformation matrix
$\mathbf{A}$	activation matrix
$\mathbf{G}$	grasp matrix
$\mathbb{S}$	skew matrix
$\mathbf{K}$	proportional gain matrix, stiffness
$\mathbf{D}$	derivative gain matrix, damping
$\mathbf{C}$	covariance matrix
t	timestep
c	cost
e	episode
k	roll-out sample
B	number of manipulators or limbs
D	number of joints
K	number of tasks
$\boldsymbol{\lambda}$	vector of weights representing soft priorities
α	scalar priority representing strict or soft relationship between two tasks
$\boldsymbol{\Psi}$	parametrization matrix for GHC and DynGHC
$\mathbb{R}$	the set of real numbers
$\|\mathbf{x}\|$	the magnitude of a vector $\mathbf{x}$
$\mathbf{x}^T$, $\mathbf{X}^T$	the transpose of a vector $\mathbf{x}$ or a matrix $\mathbf{X}$
$\mathbf{X}^{-1}$	the inverse of a matrix $\mathbf{X}$
$\mathbf{X}^{+}$	the Moore-Penrose inverse of a matrix $\mathbf{X}$
$\mathbf{X}^{\mathbf{W},+}$	the $\mathbf{W}$-weighted pseudoinverse of a matrix $\mathbf{X}$

List of Algorithms

List of Figures

List of Tables

List of References

[1] Petar Kormushev, Sylvain Calinon, and Darwin G. Caldwell. Robot Motor Skill Coordination with EM-based Reinforcement Learning. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3232–3237, 2010.

[2] Seungsu Kim, Elena Gribovskaya, and Aude Billard. Learning Motion Dynamics to Catch a Moving Object. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 106–111, 2010.

[3] Auke Jan Ijspeert, Jun Nakanishi, and Stefan Schaal. Learning attractor landscapes for learning motor primitives. *Advances in Neural Information Processing Systems*, pages 1547–1554, 2003.

[4] R. Felix Reinhart and Jochen J. Steil. Efficient policy search with a parameterized skill memory. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1400–1407, 2014.

[5] Dana Kulić, Christian Ott, Dongheui Lee, Junichi Ishikawa, and Yoshihiko Nakamura. Incremental learning of full body motion primitives and their sequencing through human motion observation. *Int. Journal of Robotics Research*, 31(3):330–345, 2012.

[6] Andre Lemme, R. Felix Reinhart, and Jochen J. Steil. Self-supervised bootstrapping of a movement primitive library from complex trajectories. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 726–732, 2014.

[7] Karim Bouyarmane and Abderrahmane Kheddar. On weight-prioritized multitask control of humanoid robots. *IEEE Transactions on Automatic Control*, 63(6):1632–1647, 2018.

[8] Oussama Kanoun, Florent Lamiraux, and Pierre-Brice Wiebe. Kinematic Control of Redundant Manipulators: Generalizing the Task-Priority Framework to Inequality Task. *IEEE Transactions on Robotics*, 27(4):785–792, 2011.

[9] Karim Bouyarmane, Joris Vaillant, Kévin Chappellet, and Abderrahmane Kheddar. Multi-robot and task-space force control with quadratic programming. *IEEE Transactions on Robotics*, 2017.

[10] Joris Vaillant, Karim Bouyarmane, and Abderrahmane Kheddar. Multi-character physical and behavioral interactions controller. *IEEE Transactions on Visualization and Computer Graphics*, 23(6):1650–1662, 2017.

[11] Oussama Khatib. A Unified Approach for Motion and Force Control of Robot Manipulators: The Operational Space Formulation. *IEEE Journal of Robotics and Automation*, 3(1):43–53, 1987.

[12] Alexander Dietrich, Thomas Wimböck, Alin Albu-Schäffer, and Gerd Hirzinger. Reactive Whole-Body Control: Dynamic Mobile Manipulation Using a Large Number of Actuated Degrees of Freedom. *IEEE Robotics Automation Magazine*, 19(2):20–33, 2012.

[13] Bruno Siciliano and Jean-Jacques Slotine. A general framework for managing multiple tasks in highly redundant robotic systems. In *Int. Conf. on Advanced Robotics*, pages 1211–1216, 1991.

[14] Luis Sentis and Oussama Khatib. Synthesis of Whole-Body Behaviors through Hierarchical Control of Behavioral Primitives. *Int. Journal of Humanoid Robotics*, 2(4):505–518, 2005.

[15] Alexander Dietrich, Christian Ott, and Alin Albu-Schäffer. An overview of null space projections for redundant, torque-controlled robots. *Int. Journal of Robotics Research*, 34(11):1385–1400, 2015.

[16] Christian Ott, Alexander Dietrich, and Alin Albu-Schäffer. Prioritized multi-task compliance control of redundant manipulators. *Automatica*, 53:416–423, 2015.

[17] Alexander Herzog, Nicholas Rotella, Sean Mason, Felix Grimminger, Stefan Schaal, and Ludovic Righetti. Momentum Control with Hierarchical Inverse Dynamics on a Torque-Controlled Humanoid. *Autonomous Robots*, 40(3):473–491, 2016.

[18] Mathieu Geisert, Andrea Del Prete, Nicolas Mansard, Francesco Romano, and Francesco Nori. Regularized hierarchical differential dynamic programming. *IEEE Transactions on Robotics*, 33(4):819–833, 2017.

[19] Enrico Mingo Hoffman, Arturo Laurenzi, Luca Muratore, Nikos G. Tsagarakis, and Darwin G. Caldwell. Multi-Priority Cartesian Impedance Control based on Quadratic Programming Optimization. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, 2018.

[20] Yoshihiko Nakamura and Hideo Hanafusa. Inverse kinematic solutions with singularity robustness for robot manipulator control. *ASME Transactions, Journal of Dynamic Systems, Measurement, and Control*, 108(3):163–171, 1986.

[21] Alexander Dietrich, Alin Albu-Schäffer, and Gerd Hirzinger. On Continuous Null Space Projections for Torque-Based, Hierarchical, Multi-Objective Manipulation. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, pages 2978–2985, 2012.

[22] Nicolas Mansard, Anthony Remazeilles, and François Chaumette. Continuity of varying-feature-set control laws. *IEEE Transactions on Automatic Control*, 54(11):2493–2505, 2009.

[23] François Keith, Pierre-Brice Wieber, Nicolas Mansard, and Abderrahmane Kheddar. Analysis of the discontinuities in prioritized tasks-space control under discreet task scheduling operations. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3887–3892, 2011.

[24] Karim Bouyarmane and Abderrahmane Kheddar. Using a multi-objective controller to synthesize simulated humanoid robot motion with changing contact configurations. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 4414–4419, 2011.

[25] Joseph Salini, Vincent Padois, and Philippe Bidaud. Synthesis of Complex Humanoid Whole-Body Behavior: A Focus on Sequencing and Tasks Transitions. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, pages 1283–1290, 2011.

[26] Federico L. Moro, Michael Gienger, Ambarish Goswami, Nikos G. Tsagarakis, and Darwin G. Caldwell. An Attractor-based Whole-Body Motion Control (WBMC) System for Humanoid Robots. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 42–49, 2013.

[27] Farhad Aghili. A unified approach for inverse and direct dynamics of constrained multibody systems based on linear projection operator: Applications to control and simulation. *IEEE Transactions on Robotics*, 21(5):834–849, 2005.

[28] Michael Mistry and Ludovic Righetti. Operational space control of constrained and underactuated systems. In *Proceedings of Robotics: Science and Systems VII*. MIT Press, 2011.

[29] Ludovic Righetti, Jonas Buchli, Michael Mistry, Mrinal Kalakrishnan, and Stefan Schaal. Optimal distribution of contact forces with inverse-dynamics control. *Int. Journal of Robotics Research*, 32(3):280–298, 2013.

[30] Farhad Aghili and Chun-Yi Su. Control of constrained robots subject to unilateral contacts and friction cone constraints. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, pages 2347–2352, 2016.

[31] Alexander Dietrich, Christian Ott, and Jaeheung Park. The Hierarchical Operational Space Formulation: Stability Analysis for the Regulation Case. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, 2018.

[32] Daniel Leidner. *Cognitive Reasoning for Compliant Robot Manipulation*. PhD thesis, University of Bremen, 2017.

[33] Donghyun Kim, Steven Jens Jorgensen, Peter Stone, and Luis Sentis. Dynamic behaviors on the nao robot with closed-loop whole body operational space control. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 1121–1128, 2016.

[34] Neville Hogan. Impedance Control: An Approach to Manipulation. I - Theory. II - Implementation. III - Applications. *ASME Transactions, Journal of Dynamic Systems, Measurement, and Control*, 107:1–24, 1985.

[35] Alin Albu-Schäffer, Christian Ott, Udo Frese, and Gerd Hirzinger. Cartesian impedance control of redundant robots: Results with the dlr-light-weight-arms. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, volume 3, pages 3704–3709, 2003.

[36] Bernd Henze, Máximo A. Roa, and Christian Ott. Passivity-based whole-body balancing for torque-controlled humanoid robots in multi-contact scenarios. *Int. Journal of Robotics Research*, 35(12):1522–1543, 2016.

[37] Jun Nakanishi, Rick Cory, Michael Mistry, Jan Peters, and Stefan Schaal. Operational space control: A theoretical and empirical comparison. *Int. Journal of Robotics Research*, 27(6):737–757, 2008.

[38] Carl D Meyer. *Matrix Analysis and Applied Linear Algebra*, volume 71. Siam, 2000.

[39] D. Goldfarb and A. Idnani. A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming*, 27(1):1–33, 1983.

[40] A. Escande, Nicolas Mansard, and Pierre-Brice Wieber. Hierarchical quadratic programming: Fast online humanoid-robot motion generation. *Int. Journal of Robotics Research*, 33(7):1006–1028, 2014.

[41] Jaemin Lee, Nicolas Mansard, and Jaeheung Park. Intermediate desired value approach for task transition of robots in kinematic control. *IEEE Transactions on Robotics*, 28(6):1260–1277, 2012.

[42] Fabrizio Flacco, Alessandro De Luca, and Oussama Khatib. Control of redundant robots under hard joint constraints: Saturation in the null space. *IEEE Transactions on Robotics*, 31(3):637–654, 2015.

[43] Mingxing Liu, Sovannara Hak, and Vincent Padois. Generalized projector for task priority transitions during hierarchical control. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, pages 768–773, 2015.

[44] T. Mergner. A neurological view on reactive human stance control. *Annual Reviews in Control*, 34(2):177–198, 2010.

[45] A. S. Deo and Ian D. Walker. Overview of damped least-squares methods for inverse kinematics of robot manipulators. *Journal of Intelligent and Robotic Systems*, 14(1):43–68, 1995.

[46] Luis Sentis, Jaeheung Park, and Oussama Khatib. Compliant control of multicontact and center-of-mass behaviors in humanoid robots. *IEEE Transactions on Robotics*, 26(3):483–501, 2010.

[47] Richard M. Murray, Zexiang Li, and S. Shankar Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC press, 1994.

[48] Masaru Uchiyama and Pierre Dauchez. Symmetric kinematic formulation and non-master/slave coordinated control of two-arm robots. *Advanced Robotics*, 7(4):361–383, 1992.

[49] Sebastian Erhart and Sandra Hirche. Internal force analysis and load distribution for cooperative multi-robot manipulation. *IEEE Transactions on Robotics*, 31(5):1238–1243, 2015.

[50] Hsiu-Chin Lin, Joshua Smith, Keyhan Kouhkiloui Babarahmati, Niels Dehio, and Michael Mistry. A projected inverse dynamics approach for dual-arm cartesian impedance control. *arxiv*, 2017, accepted for ICRA 2018.

[51] Christian Ott, Maximo A. Roa, and Gerd Hirzinger. Posture and balance control for biped robots based on contact force optimization. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 26–33, 2011.

[52] Tamim Asfour, Júlia Borràs, Christian Mandery, Peter Kaiser, Eren Erdal Aksoy, and Markus Grotz. *On the Dualities Between Grasping and Whole-Body Loco-Manipulation Tasks*, pages 305–322. Springer International Publishing, Cham, 2018.

[53] Luis Sentis and O. Khatib. A Whole-Body Control Framework for Humanoids Operating in Human Environments. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, pages 2641–2648, 2006.

[54] Valerio Ortenzi, Maxime Adjigble, Jeffrey A. Kuo, Rustam Stolkin, and Michael Mistry. An experimental study of robot control during environmental contacts based on projected operational space dynamics. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 407–412, 2014.

[55] Niels Dehio, R. Felix Reinhart, and Jochen J. Steil. Multiple Task Optimization with a Mixture of Controllers for Motion Generation. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 6416–6421, 2015.

[56] N. Dehio, R. Felix Reinhart, and Jochen J. Steil. Continuous Task-Priority Rearrangement during Motion Execution with a Mixture of Torque Controllers. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 264–270, 2016.

[57] Nikolaus Hansen and Andreas Ostermeier. Completely Derandomized Self-Adaptation in Evolution Strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[58] Freek Stulp and Olivier Sigaud. Robot Skill Learning: From Reinforcement Learning to Evolution Strategies. *Paladyn, Journal of Behavioral Robotics*, 4(1):49–61, 2013.

[59] Nikolaus Hansen. Homepage: CMA-ES documentation with source code. `https://www.lri.fr/~hansen` (Date last accessed on 22.02.2016).

[60] Nicolas Docquier, Antoine Poncelet, and Paul Fisette. ROBOTRAN: a powerful symbolic gnerator of multibody models. *Mechanical Sciences*, 4(1):199–219, 2013.

[61] Houman Dallali, Mohamad Mosadeghzad, Gustavo A Medrano-Cerda, Nicolas Docquier, Petar Kormushev, Nikos Tsagarakis, Zhibin Li, and Darwin Caldwell. Development of a Dynamic Simulator for a Compliant Humanoid Robot Based on a Symbolic Multibody Approach. In *IEEE Int. Conf. on Mechatronics*, pages 598–603, 2013.

[62] Auke Jan Ijspeert, Jun Nakanishi, Heiko Hoffmann, Peter Pastor, and Stefan Schaal. Dynamical Movement Primitives: Learning Attractor Models for Motor Behaviors. *Neural Computation*, 25(2):328–373, 2013.

[63] Ryan Lober, Vincent Padois, and Olivier Sigaud. Multiple Task Optimization using Dynamical Movement Primitives for Whole-Body Reactive Control. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 193–198, 2014.

[64] Katharina Muelling, Jens Kober, and Jan Peters. Learning Table Tennis with a Mixture of Motor Primitives. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 411–416, 2010.

[65] Valerio Modugno, Gerard Neumann, Elmar Rueckert, Giuseppe Oriolo, Jan Peters, and Serena Ivaldi. Learning soft task priorities for control of redundant robots. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, pages 221–226, 2016.

[66] Valerio Modugno, Ugo Chervet, Giuseppe Oriolo, and Serena Ivaldi. Learning soft task priorities for safe control of humanoid robots with constrained stochastic optimization. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 101–108, 2016.

[67] R. Lober, Vincent Padois, and Olivier Sigaud. Variance Modulated Task Prioritization in Whole-Body Control. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3944–3949, 2015.

[68] João Silvério, Sylvain Calinon, Leonel Rozo, and Darwin G. Caldwell. Learning competing constraints and task priorities from demonstrations of bimanual skills. *arxiv*, 2017.

[69] João Silvério, Yanlong Huang, Leonel Rozo, Sylvain Calinon, and Darwin G. Caldwell. Probabilistic Learning of Torque Controllers from Kinematic and Force Constraints. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2018.

[70] Sylvain. Calinon and Dongheui Lee. Learning control. In P. Vadakkepat and A. Goswami, editors, *Humanoid Robotics: a Reference*. Springer, 2018.

[71] Niels Dehio, Daniel Kubus, and Jochen J. Steil. Continuously Shaping Projections and Operational Space Tasks. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 5995–6002, 2018.

[72] Enrico Simetti and Giuseppe Casalino. A novel practical technique to integrate inequality control objectives and task transitions in priority based control. *Journal of Intelligent & Robotic Systems*, 84(1):877–902, 2016.

[73] Oliver Brock, Oussama Khatib, and Sriram Viji. Task-consistent obstacle avoidance and motion behavior for mobile manipulation. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, volume 1, pages 388–393, 2002.

[74] S. S. Kim and M. J. Vanderploeg. QR decomposition for state space representation of constrained mechanical dynamic systems. *Journal of Mechanisms, Transmissions, and Automation in Design*, 108(2):183–188, 1986.

[75] Amir Fijany and Antal K. Bejczy. Efficient jacobian inversion for the control of simple robot manipulators. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, volume 2, pages 999–1007, 1988.

[76] Mingxing Liu, Yang Tan, and Vincent Padois. Generalized hierarchical control. *Autonomous Robots*, 40(1):17–31, 2016.

[77] Mingxing Liu, Ryan Lober, and Vincent Padois. Whole-body hierarchical motion and force control for humanoid robots. *Autonomous Robots*, 40(3):493–504, 2016.

[78] Nicolas Mansard and François Chaumette. Task sequencing for high-level sensor-based control. *IEEE Transactions on Robotics*, 23(1):60–72, 2007.

[79] Nicolas Mansard, Oussama Khatib, and Abderrahmane Kheddar. A unified approach to integrate unilateral constraints in the stack of tasks. *IEEE Transactions on Robotics*, 25(3):670–685, 2009.

[80] Fabrizio Flacco, Alessandro De Luca, and Oussama Khatib. Motion control of redundant robots under joint constraints: Saturation in the null space. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, pages 285–292, 2012.

[81] João Silvério, Sylvain Calinon, Leonel Rozo, and Darwin G. Caldwell. Learning task priorities from demonstrations. *arxiv*, 2018.

[82] Tadej Petrič and Leon Žlajpah. Smooth continuous transition between tasks on a kinematic control level: Obstacle avoidance as a control problem. *Robotics and Autonomous Systems*, 61(9):948–959, 2013.

[83] Fabrizio Flacco and Alessandro De Luca. Unilateral constraints in the reverse priority redundancy resolution method. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2564–2571, 2015.

[84] Sang-ik An and D. Lee. Prioritized inverse kinematics with multiple task definitions. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, pages 1423–1430, 2015.

[85] Gerardo Jarquín, Gustavo Arechavaleta, and Vicente Parra-Vega. Time parametrization of prioritized inverse kinematics based on terminal attractors. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1612–1617, 2011.

[86] Hyejin Han, Jaemin Lee, and Jaeheung Park. A continuous task transition algorithm for operational space control framework. In *Int. Conf. on Ubiquitous Robots and Ambient Intelligence*, pages 148–152, 2012.

[87] Hyejin Han and Jaeheung Park. Robot Control near Singularity and Joint Limit Using a Continuous Task Transition Algorithm. *Int. Journal of Advanced Robotic Systems*, 10, 2013.

[88] Gerardo Jarquín, Adrien Escande, Gustavo Arechavaleta, Thomas Moulard, Eiichi Yoshida, and Vicente Parra-Vega. Real-time smooth task transitions for hierarchical inverse kinematics. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 528–533, 2013.

[89] Gerardo Jarquín, Gustavo Arechavaleta, and Vicente Parra-Vega. Continuous kinematic control with terminal attractors for handling task transitions of redundant robots. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, pages 1976–1981, 2013.

[90] Gustavo Arechavaleta, América Morales-Díaz, Héctor M. Pérez-Villeda, and Mario Castelán. Hierarchical task-based control of multirobot systems with terminal attractors. *IEEE Transactions on Control Systems Technology*, 25(1):334–341, 2017.

[91] Michael Mistry, Jun Nakanishi, Gordon Cheng, and Stefan Schaal. Inverse kinematics with floating base and constraints for full body humanoid robot control. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 22–27, 2008.

[92] Jerry Pratt, Chee-Meng Chew, Ann Torres, Peter Dilworth, and Gill Pratt. Virtual model control: An intuitive approach for bipedal locomotion. *Int. Journal of Robotics Research*, 20(2):129–143, 2001.

[93] Robert Haschke, Jochen J. Steil, I. Steuwer, and Helge Ritter. Task-oriented quality measures for dextrous grasping. In *Int. Symposium on Computational Intelligence in Robotics and Automation*, pages 689–694, 2005.

[94] Alessandro Marino, Giuseppe Muscio, and Francesco Pierri. Distributed cooperative object parameter estimation and manipulation without explicit communication. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, pages 2110–2116, 2017.

[95] Dennis Leroy Wigand, Arne Nordmann, Niels Dehio, Michael Mistry, and Sebastian Wrede. Domain-Specific Language Modularization Scheme Applied to a Multi-Arm Robotics Use-Case. *Journal of Software Engineering for Robotics*, 8(1):45–64, 2017.

[96] Hsiu-Chin Lin, Joshua Smith, Keyhan Kouhkiloui Babarahmati, Niels Dehio, and Michael Mistry. A projected inverse dynamics approach for multi-arm cartesian impedance control. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, pages 1–5, 2018.

[97] Niels Dehio, Joshua Smith, Dennis Leroy Wigand, Guiyang Xin, Hsiu-Chin Lin, Jochen J. Steil, and Michael Mistry. Modeling and Control of Multi-Arm and Multi-Leg Robots: Compensating for Object Dynamics during Grasping. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, pages 294–301, 2018.

[98] Ludovic Righetti, Jonas Buchli, Michael Mistry, and Stefan Schaal. Inverse dynamics control of floating-base robots with external constraints: a unified view. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, pages 1085–1090, 2011.

[99] João Silvério, Leonel Rozo, Sylvain Calinon, and Darwin G. Caldwell. Learning bimanual end-effector poses from demonstrations using task-parameterized dynamical systems. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 464–470, 2015.

[100] Marco Hutter, Christian Gehring, Dominic Jud, Andreas Lauber, C. Dario Bellicoso, Vassilios Tsounis, Jemin Hwangbo, Karen Bodie, Peter Fankhauser, Michael Bloesch, Remo Diethelm, Samuel Bachmann, Amir Melzer, and Mark Hoepflinger. Anymal - a

highly mobile and dynamic quadrupedal robot. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 38–44, 2016.

[101] Michele Focchi, Andrea del Prete, Ioannis Havoutis, Roy Featherstone, Darwin G. Caldwell, and Claudio Semini. High-slope terrain locomotion for torque-controlled quadruped robots. *Autonomous Robots*, 41(1):259–272, 2017.

[102] Tsuneo Yoshikawa. Virtual truss model for characterization of internal forces for multiple finger grasps. *IEEE Transactions on Robotics and Automation*, 15(5):941–947, 1999.

[103] Yisoo Lee, Soonwook Hwang, and Jaeheung Park. Balancing of humanoid robot using contact force/moment control by task-oriented whole body control framework. *Autonomous Robots*, 40(3):457–472, 2016.

[104] Christopher G. Atkeson, Chae H. An, and John M. Hollerbach. Estimation of inertial parameters of manipulator loads and links. *Int. Journal of Robotics Research*, 5(3):101–119, 1986.

[105] Fabrizio Caccavale, Pasquale Chiacchio, Alessandro Marino, and Luigi Villani. Six-dof impedance control of dual-arm cooperative manipulators. *IEEE Transactions on Mechatronics*, 13(5):576–586, 2008.

[106] Allison M. Okamura, Niels Smaby, and Mark R. Cutkosky. An overview of dexterous manipulation. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, volume 1, pages 255–262, 2000.

[107] Thomas Wimbock, Christian Ott, and Gerd Hirzinger. Analysis and experimental evaluation of the intrinsically passive controller (ipc) for multifingered hands. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, pages 278–284, 2008.

[108] Bojan Nemec, Nejc Likar, Andrej Gams, and Aleš Ude. Bimanual human robot cooperation with adaptive stiffness control. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 607–613, 2016.

[109] Antonio Paolillo, Kevin Chappellet, Anastasia Bolotnikova, and Abderrahmane Kheddar. Interlinked visual tracking and robotic manipulation of articulated objects. *IEEE Robotics and Automation Letters*, 3(4):2746–2753, 2018.

[110] Borhan Beigzadeh, Majid Nili Ahmadabadi, Ali Meghdari, and Adel Akbarimajd. A dynamic object manipulation approach to dynamic biped locomotion. *Robotics and Autonomous Systems*, 56(7):570–582, 2008.

[111] Aaron M. Johnson and Daniel E. Koditschek. Legged self-manipulation. *IEEE Access*, 1:310–334, 2013.

[112] Guiyang Xin, Hsiu-Chin Lin, Joshua Smith, Oguzhan Cebe, and Michael Mistry. A Model-based Hierarchical Controller for Legged Systems subject to External Disturbances. In *IEEE/RSJ Int. Conf. on Robotics and Automation*, 2018.

[113] Niels Dehio and Jochen J. Steil. A Comparison of Null-space Projection and Mixture of Torque Controllers for Motion Generation. In *Proc. 9th Int. Workshop on Human-Friendly Robotics*, 2016.

[114] Niels Dehio and Jochen J. Steil. Optimization of Manipulator Base Placements for Multi-Arm Grasping. In *Proc. Int. Workshop on Examining Sensing Modalities for Robust and Dexterous Object Manipulation*, 2018.